

المؤسسة العامة للدراسات والنشر والتوزيع ش.م.م



جاك ريفيير

البرمجة : لغة المؤول (الاسمبلر)

ترجمة د. عبد الحسن الحسيني



البرصة بلغة المؤلف
(الاسمبار)

جميع الحقوق محفوظة

الطبعة الأولى

1410 هـ - 1990 م

دار المؤسسة العامة للدراسات والنشر والتوزيع

بيروت - الحفراء - شارع النيل اده - ساحة سلاء

هاتف : ٨٠٢٤٢٨ - ٨٠٢٤٠٧ - ٨٠٢٢٩٦

بيروت - القصية - ساحة طاهر هاتف : ٣٠١٠٣٠ - ٣١١٣١٠

ص. ب. ٦٣١١ / ١١٣ مكتب : ٢٠٦٦٥١.F - ٢٠٦٦٨٠ - لبنان

سلسلة بإشراف
د. عبد الحسن الحسيني

جاك ريفيير

البرمجة بلغة المؤول (الاسمبلر)

ترجمة د. عبد الحسن الحسيني

 المؤسسة الجامعة للدراسات والنشر والتوزيع

هذا الكتاب ترجمة :

**LA PROGRAMMATION
EN ASSEMBLEUR**

Par

Jacques RIVIERE

© BORDAS, PARIS

تقديم

تعتبر لغة أسيمبلر (المؤول) من اللغات الفعالة وذات الإمكانيات الكبيرة نظراً لأنها تسمح للمبرمج باستعمال جميع إمكانيات ومقدرات وموارد الحاسب ، كما تسمح له بالدخول إلى « قلب » الآلة والعمل بالمراسف الداخلية للحاسب ، مما يضيفي على البرنامج المكتوب بهذه اللغة فعالية كبيرة خصوصاً فيما يتعلق بالدقة والسرعة والعمل في الوقت الفعلي (real time) المستعمل كثيراً لإدارة العمليات الصناعية .

هذا الكتاب يُعالج لغة أسيمبلر الخاصة بعائلة الحاسبات IBM 360/370 التي شهدت إنتشاراً واسعاً في حقل المعلوماتية وأحدثت ثورة في صناعة الحاسبات في السنوات الأخيرة وبقيت تركيبة وهيكلية هذه الآلات مُستعملة وصالحة في وقتنا هذا وجرى إستعمالها والإفادة منها حتى في صناعة المعالج الصغري وتصميم الميكرو حاسبات .

وبالنسبة للبرمجة بلغة المؤول ، فإن تقنية هذه البرمجة لا تختلف أبداً من آلة إلى أخرى ، صغيرة كانت أم كبيرة ، معالجاً صغيراً أو نظاماً كبيراً . أما الفرق الوحيد فيمكن في كون كود - الآلة يختلف من آلة إلى أخرى ، أما طريقة العمل والمعالجة وإستعمال المراسف والذاكرة فلا تختلف إلا في عدد المراسف المبلوغة من المبرمج ، وبالتالي فإن التصرف على أي مؤول يبقى صالحاً بالنسبة لمعالج آخر بمؤول آخر .

وهنا يجب الإشارة إلى أن مؤول IBM/370 يتألف من أكبر عدد ممكن من التعليقات، وعدد مراسف الحاسب يعادل 16 للمعطيات و16 للعناوين ويستعمل عدداً كبيراً من طرق العنونة ، يصلح قسم منها لعنونة المعلومات عند إستعمال المعالج الصغري .

المترجم

تمهيد

لماذا كتاب جديد يختص بلغة المؤول (Assembler) ؟ وما هو المؤول ؟ هل تعرفون
مُبرمجين يعملون بلغة المؤول حتى الآن ، بينها تقدّم اللغات المتطورة إمكانيات وتسهيلات
جديدة ؟

كثيراً ما نسمع جميع هذه الأسئلة إضافة إلى أخرى مدهشة ، ولن نحاول هنا في
هذا التمهيد أن نجاب عنها ، السؤال بعد الآخر ، ولكن سنحاول توضيح هدفنا من
هذا الكتاب .

وُضع هذا الكتاب بسبب ثلاث ملاحظات :

- إن إتقان لغة المؤول هو الطريقة الأفضل لفهم طريقة عمل الحاسب .
- بواسطة إتقان لغة المؤول ، مهما يكن ، سنستطيع التفكير بسهولة أكثر وإدراك ماذا
يحدث عندما نعمل بلغة أكثر تطوراً ، والبحث عن الأخطاء سيكون أكثر سهولة .
- عند نزول الميكروبروسور إلى الأسواق ، أليس من الأفضل إتقان هذه اللغة الموجودة
على هذه الآلات الصغيرة ؟ مع الإشارة إلى أن المؤول يبقى الوسيلة الفضلى لإنشاء
وخلق المناهج الجديدة .

هكذا فلكتابنا هذا هدف تربوي . وهو ليس عبارة عن كتاب مساعد ومرجع في
المعنى الذي نفهمه من المرجع المساعد الخاص بالمنتج ، ولكنه عبارة عن مساعد كافٍ
وكامل لفهم عمليات الإنشاء والبرمجة المهمة .

وهو موجّه إلى أولئك الراغبين بفهم طريقة عمل الآلات التي يستعملونها . ولقد
حاولنا الإجابة عن المسائل التي ستواجهنا ، وبشكل خاص لدى الطلاب الذين يرغبون
بمعرفة لغة المؤول بعد معرفتهم بإحدى اللغات المتطورة . وهذا هو دور الفصل الأول من
الكتاب الذي يحتوي على عرض لتركيب وطريقة عمل الحاسب ، وهذا العرض جرى من
خلال تفكير بسيط يتعلق بآلة ذات استعمال كبير : الحاسب الجيبى . ولأجل هؤلاء

أيضاً قمنا بعرض مشاكل العنونة ، التقطيع ، تنقيح الأربطة (link editor) ، الشحن (loading) ، والإنقطاعات عند الإدخال والإخراج (I/O interruption) .

وهو موجه أيضاً الى كل من يرغب بالعمل بلغة المؤول ، إما على الآلة المعتمدة كمرجع وهي الحاسب IBM 370 ، أو على الحاسب الشخصي الميكروكوميبيوتر . وهنا نؤكد بأن جميع لغات التأويل هي متشابهة بشكل نستطيع معه بعد معرفة مؤول معين أن نتكيف بسهولة للعمل على مؤول آخر بآلة أخرى ، ولهذا الهدف قمنا بإضافة مسائل بسيطة ، نجد التطبيق العملي لها على أغلب الحاسبات .

وفي النهاية ، لهؤلاء الذين يعرفون المؤول ، قمنا بإثبات الإمكانيات التي يُقدمها التأويل المشروط وإستعمال الماكرو تعليمات (MACRO INSTRUCTIONS) . ونصائح هذا الكتاب التي تدور حول البرمجة الجيدة هي عبارة عن عناصر للتفكير يصبح في نهايتها البرنامج مختلفاً عن تلك المجموعة من التعليمات المبهمة كما في اللغة الثنائية . ومن الممكن إنشاء وتركيب برنامج مكتوب بلغة المؤول بشكل يصبح معه واضحاً كوضوح برنامج بلغة كوبول .

لماذا جرى إختيار الحاسب IBM 370 ؟

- لأنها شاملة وعامة . وأكثر صيغ لغة المؤول العاملة عليها جرى إستعمالها وتطويرها من قبل جميع المنتجين والصانعين .

- لماضيها ومستقبلها : إن المواصفات الخاصة بهذه اللغة والتي جهزت مع النظام IBM 360 ، قد جرت المحافظة عليها في الحاسبات IBM 370 وفي الأنظمة الجديدة من السلسلة 3000 و4000 إضافة إلى أغلب حاسبات IBM الجديدة .

عموميات

1. الآلة البسيطة

هذا الفصل الأول هو مخصص للمبتدئين . أما الذي يتمتع بمفاهيم كافية تتعلّق بهيكل المكنة فيمكنه أن يبدأ دراسته من الفصل الثاني . إلّا أننا نعتقد بأنه يعرض ويوضح النقاط الأساسية لعملية الفهم اللاحقة . وهو يعرف المصطلحات الأساسية المتعلقة بدورة تنفيذ تعليمات الآلة .

1.1 . دراسة للآلة الحاسبة الصغيرة الجيبية

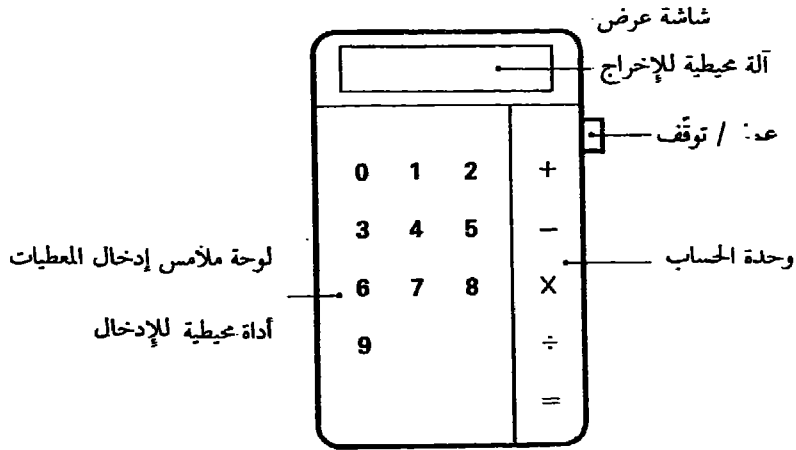
منذ النظرة الأولى ، تبدو الآلة الحاسبة الجيبية وكأنها مؤلفة من العناصر التالية :

- زر للعمل / ولوقف العمل .
- لوحة ملامس رقمية .
- شاشة للعرض .
- مجموعة من ملامس التحكم + ، - ، = ،
- فلنقم بعملية حساب بسيطة ، القسمة مثلاً . عملية المعالجة ستجري كما يلي :
- 1- وضع الآلة الحاسبة في العمل .
- 2- ادخال العدد الأول (المقسوم) وعرضه .
- 3- ضغط الزر الخاص بالقسمة .
- 4- إدخال العدد الثاني (القاسم) وعرضه .
- 5- الضغط على الزر = ، وعرض النتيجة .
- 6- إيقاف عمل الآلة الحاسبة .

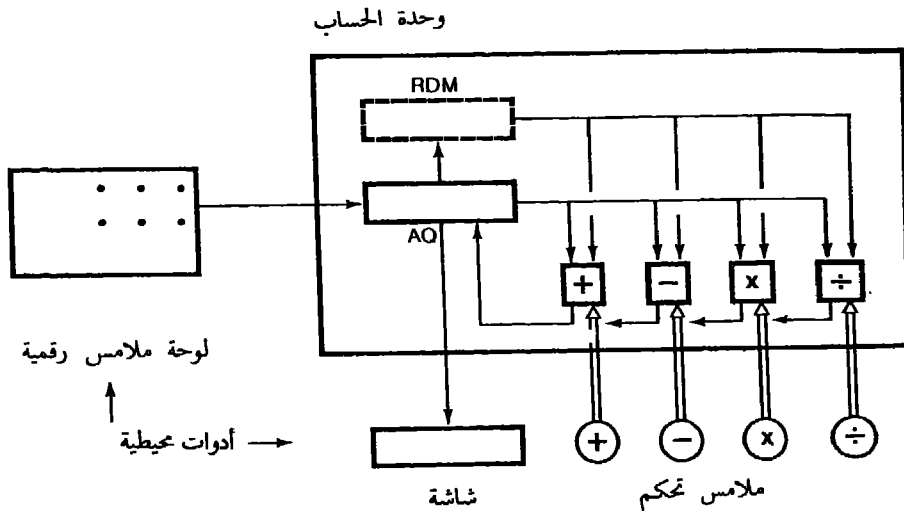
هذه السلسلة من العمليات تتطلب بعض الملاحظات :

- ترتيب العمليات هو مُحدّد وثابت ؛ لا يمكن عكس العمليات 2 و 4 .
- تتمتع مكتتنا ، إضافة إلى الدالة حساب (Compute) ، بدالة (مهمّة) لإدخال المعطيات وبدالة لإخراج المعطيات (العرض على الشاشة) .
- عند إجراء العملية رقم 4 ، يختفي العدد المعروض على الشاشة ، قبل أن تتم عملية القسمة (يجب أن نعطي الصلاحية للعملية بالضغط على الملامس =) ، يجب إذاً ،

وبشكل إلزامي ، أن تحتوي المكنة على ذاكرة يُحزَّن فيها العدد الأول بانتظار نهاية إدخال القاسم . فلنعرض المخطط التوضيحي⁽¹⁾ :



مخطط 1.1



مخطط 2.1

(1) إن المخططات المعروضة في هذا الفصل لا تدعي تمثيل الدقة التكنولوجية ولكنها تعرض فقط الدالات الأساسية المفيدة للمبرمج .

هذا المخطط يُميز بين نوعين من الخطوط . الخطوط البسيطة (→) والتي تناسب خطوط إنتقال المعطيات والخطوط المزدوجة (⇒) والتي تناسب خطوط تنقل الأوامر .

تعريفات :

نسَمي وحدة حساب مجموعة دارات الجمع والطرح ، ... تُخزّن معطيات الحساب في المناطق RDM و AQ والتي تُدعى مراصف (register) . المرصف RDM يُستخدم لتخزين العدد الأول الداخِل إلى AQ للسماح بإدخال العدد الثاني .

نتيجة الحساب توضع دائماً في مرصف خاص AQ ولذلك نطلق عليه إسم مركم (Accumulator) . أمّا لوحة الملامس الرقمية وشاشة العرض فنطلق عليها الإسم : الأدوات المحيطة للإدخال والإخراج (I/O peripherals) .

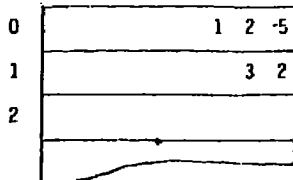
2.1 . دراسة حاسبة جيبيّة مع ذاكرة

لنضيف الى الحاسبة الجيبية مجموعة من خلايا الذاكرة التي سنطلق عليها الإسم : ذاكرة مركزية (Central memory) . كل خلية من الذاكرة ، وتدعى أيضاً كلمة - آية (machine word) ، يمكنها كالمراصف أن تحتوي على مخططات أو على نتائج الحساب . إلى كل خلية سربط عدداً محدداً يُدعى عنوان الخلية ويسمح بتمييز الخلايا فيما بينها . المؤثرات الأساسية (+ ، - ، ...) هي عبارة عن مؤثرات ثنائية (نقصد بذلك أنها تجري بين متأثرين (operators)) . أحد المتأثرين يكون موجوداً في المرصف AQ والآخر في المرصف RDM (مرصف معطيات الذاكرة) . كما في الحاسبات البسيطة فإن النتيجة ستكون موجودة في AQ . يصبح من الضروري أن يكون بتصرفنا :

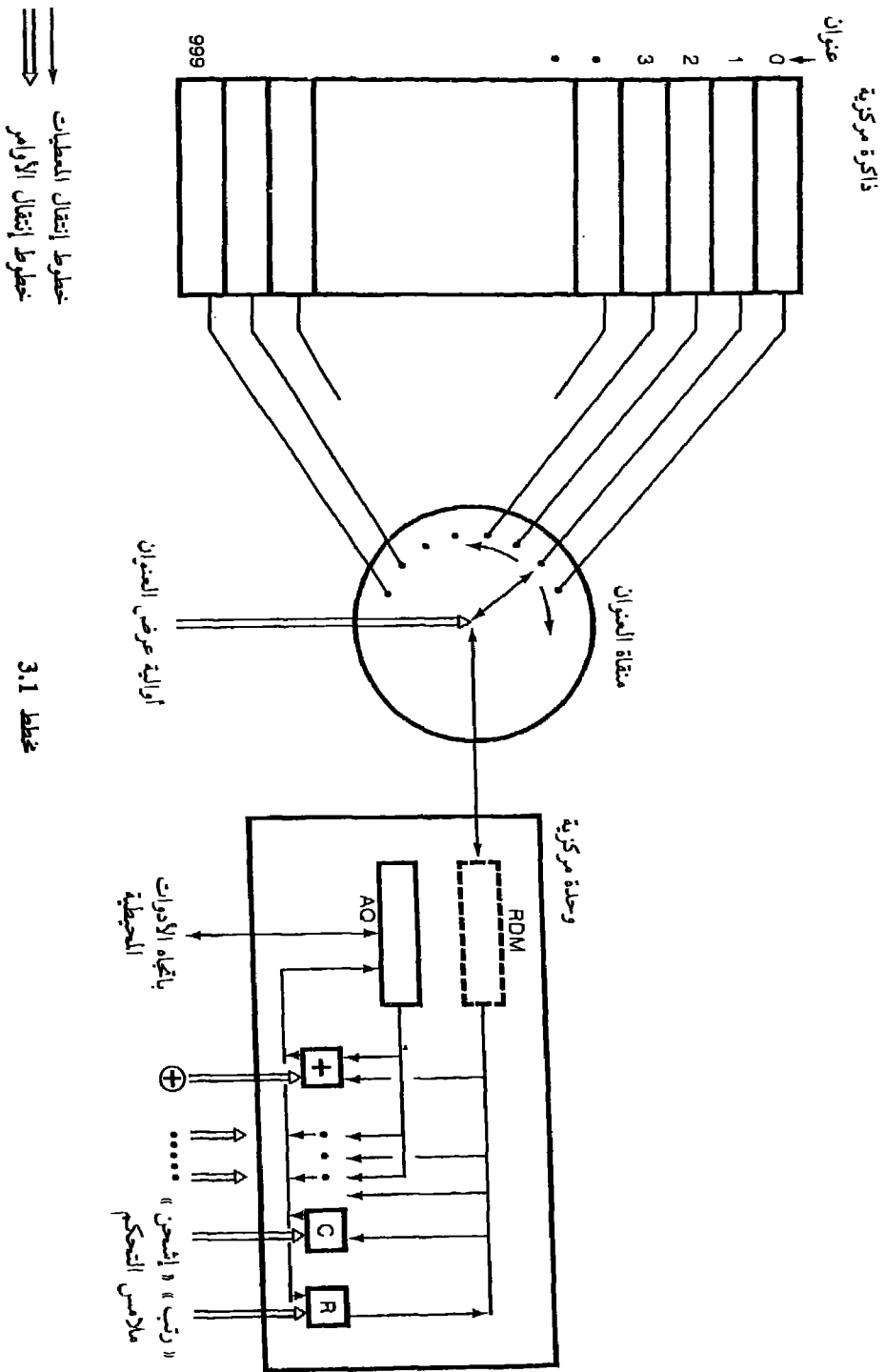
- نظام لإختيار العنوان الذي يؤمن الإتصال بين إحدى خلايا الذاكرة والمرصف RDM ؛

- دارتان إضافيتان للشحن والترتيب ، لشحن مضمون خلية من الذاكرة في المركم وترتيب مضمون المركم في عنوان معيّن . هكذا دارات هي موجودة على جميع الحاسبات الجيبية وتتمتع بخلية ذاكرة واحدة على الأقل . نخطط حاسبة كهذه هو ممثّل على الشكل 3.1 .

إنّ منقاة العنوان هي هنا موضحة بواسطة ملهاس دائري يؤمن الإتصال بين خلية من الذاكرة بعنوان معيّن ومضمون المرصف RDM . ويتعلّق إنجاء إنتقال المعطيات بالمؤثر أو بالإشارة الحاسوبية المعتمدة .



مثال حول عملية حساب بسيطة .
لنفترض، إن الذاكرة تحتوي على المعطيات التالية :



نرغب بجمع مضمون الخلية ذات العنوان 0 مع مضمون الخلية ذات العنوان 1 وبوضع النتيجة في العنوان 2 . فلنستعمل الترميز الكلاسيكي : (ALPHA) ، حيث ALPHA هي عبارة عن عنوان ، يشير الى مضمون الخلية ذات العنوان ALPHA . هكذا فإن (0) يعني هنا القيمة 125 . السهم سيؤدي إلى اتجاه انتقال المعطيات : AQ → (0) يعني تخزين مضمون الخلية ذات العنوان (0) في المرمز AQ ، أي تخزين العدد 125 في AQ .

لإجراء عملية الحساب يجب :

- 1- تركيز منقاة العنوان على 0 والضغط على الزر «إشحن» ، مما يؤدي إلى تنفيذ العملية : AQ → (0) .
 - 2- تركيز منقاة العنوان على 1 والضغط على الزر + .
- هذا يسمح بإجراء العملية $AQ + (1) \rightarrow AQ$. هكذا فإن هذه العملية يمكن تقسيمها إلى إثنين .

$$(1) \rightarrow RDM \quad (أ)$$

$$AQ + RDM \rightarrow AQ \quad (ب)$$

- 3- تركيز منقاة العنوان على 2 والضغط على الزر «خزن» . هذا ما يسمح بتنفيذ العملية (2) : AQ → .

في نهاية هذه العمليات ، ستحتوي الخلية ذات العنوان 2 على العدد 157 . والمرصف AQ يحتوي على القيمة النهائية .

ملاحظات :

جميع عمليات الحساب تتم بين المراسف AQ و RDM وليس من الذاكرة إلى الذاكرة . وهذا ما يؤدي إلى الحاجة إلى إجراء عملية شحن مسبقة للمرمز . المراسف هي إذاً عبارة عن ذاكرة مرتبطة مباشرة بدارات الحساب .

للإشارة إلى مضمون خلايا الذاكرة سنعتمد على الترميز (عنوان adresse) بشكل نستطيع معه تمييز العنوان عن مضمونه ، أي إسم «nom» الخلية وقيمته . المراسف المذكورة لا ترد داخل أهلة لأنه لا يوجد أي خلط ممكن بين المضمون والإسم : نعود دائماً إلى مضمون المراسف .

- 3.1 . من الحاسبة الصغيرة إلى الحاسب الكبير (الكومبيوتر)
- إن كل معالجة تتناول معطيات وتسلسلاً دقيقاً من الأفعال ، والأوامر على الملامس + ، - ، ونوع الحاسبة المعتمدة حتى الآن لا يسمح بتخزين معطيات المسألة .

الفرق الأكبر بين الحاسبة ذات الذاكرة والحاسب الكبير يكمن في كون الأخير :
 - يُخزّن ليس فقط المعطيات ولكن الأوامر المطلوب إجراؤها على المعطيات .
 - يتمتع بأولية لربط الأوامر التي ستسمح له بتنفيذ هذه الأوامر حسب الترتيب الواردة فيه . هكذا ، فذاكرة الحاسب المركزية (C.M) ستحتوي على معطيات المسألة وطريقة معالجتها للحصول على النتائج .

تعريفات :

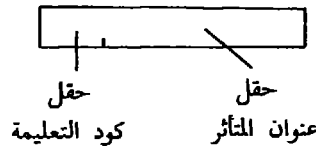
في البداية ، سنعني كلمة أمر (Command) بالتعليمية (instruction) أو التعليمية الآلية (machine instruction) . ومجموعة التعليمات والمعطيات المرتبطة بها تؤلف البرنامج . أما الملامس + ، - فستختفي . ويصبح عندئذٍ من البديهي أن لا يعمل الحاسب إلا إذا كان البرنامج مسجلاً في ذاكرته المركزية .

1.3.1 - هيكلية التعليمات الآلية

حسب المثل المذكور أعلاه في الفقرة 2.1 ، نستطيع أن نقول أن التعليمات الآلية هي مؤلفة من معلومتين :

- 1- رقم يدل على الدارة المعتمدة من اوحدة المركزية .
- 2- رقم يدل على عنوان المتأثر (Operand) .

إذا كانت التعليمية تعمل بمتأثرين (الحالة + ، - ، ...) ، يكون المتأثر الأول مشحوناً مسبقاً في المرمك (ACC) . هاتان المعلومتان ستكونان موجودتين في كلمة من الذاكرة بشكل مكوّد رقمياً ، مثلاً حسب الطريقة التالية :



وستسمح أولية تكويد التعليمية ، التي سنقوم بتوضيحها لاحقاً ، بكشف ومعرفة الفعل المطلوب إجراؤه على المتأثر الموجود على العنوان المذكور في التعليمية .
 مثلاً :

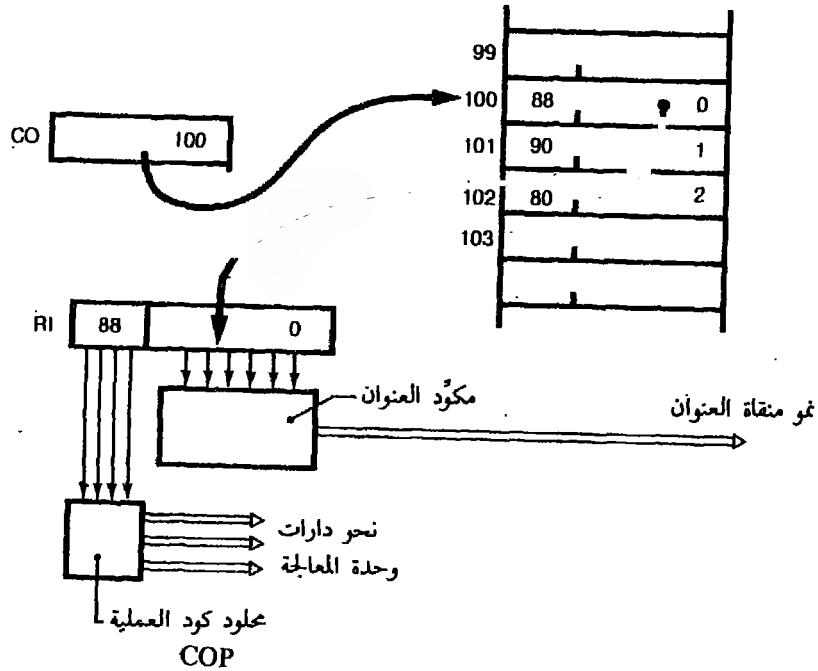
لنفترض بأن كود عملية الشحن COP هو 88 ، وإن كود الجمع هو 90 وكود الخزن هو 80 . فلنخزّن البرنامج الذي يقوم بجمع الخليتين 0 و 1 مع وضع النتيجة على العنوان 2 ، بدءاً من العنوان 100 . نحصل عندئذٍ على صورة الذاكرة التالية :

99			
100	8	8	0
101	9	0	1
102	8	0	2
103			

تنفيذ البرنامج يفترض ربطاً متتالياً للتعليمات الموجودة ، بدءاً من العنوان 100 ثم 101 ، ...

2.3.1 . أولية معرفة وربط التعليمات

تحتوي الذاكرة على نوعين من المعلومات بطبيعة دلالية مختلفة . المعطيات والتعليمات . من الضروري معاينة ومعرفة الخلية التي تحتوي على التعليمات المطلوب تنفيذها . لهذا الهدف ، هناك مرصف خاص يسمى العدّاد الرئيسي الترتيبي (CO) أو عدّاد البرنامج program counter الذي سيحتوي في كل لحظة على العنوان التالي للتعليمات المطلوب تنفيذها . وبشكل خاص ، وفي البداية ، سيكون مشحوناً بعنوان أول تعليمات .



4.1 خطط

منذ اللحظة التي يحتوي فيها CO على عنوان التعليم ، فإن دورة التنفيذ تبدأ :

1 - إرسال التعليم التي يشير إليها عداد البرنامج إلى مصرف التعليم RI المرتبط بمكود للعملية COP وبنقطة العنوان .

2 - تكوين العنوان الذي يقوم بتركيز منقاة العنوان ، وبحلود (يفك كود) COP الذي يضع الدارة المناسبة من وحدة المعالجة في حالة العمل .

3 - تنفيذ العملية المطلوبة بواسطة وحدة المعالجة التي ستصبح في طور العمل .
خلال المرحلة الثانية لن يكون من الضروري أن يؤثر CO على التعليم الموجودة في طور التنفيذ ، وخلال هذه المرحلة إذا تزداد قيمة عداد البرنامج CO واحداً (1) ليؤثر على التعليم التالية المطلوب تنفيذها .

بعد تنفيذ التعليم ، يعود الحاسب الى المرحلة الأولى بالقيمة الجديدة لعداد البرنامج CO وهذا يتتابع حتى نلتقي تعليم خاصة بوقف البرنامج .
يبقى أن نشير إلى مختلف مراحل التنفيذ هي متزامنة بواسطة نبضات ساعة داخلية .

المخطط 5.1 التالي يعرض لمختلف المهام التي درستها . وهو يشكل المخطط العملي للحاسب .

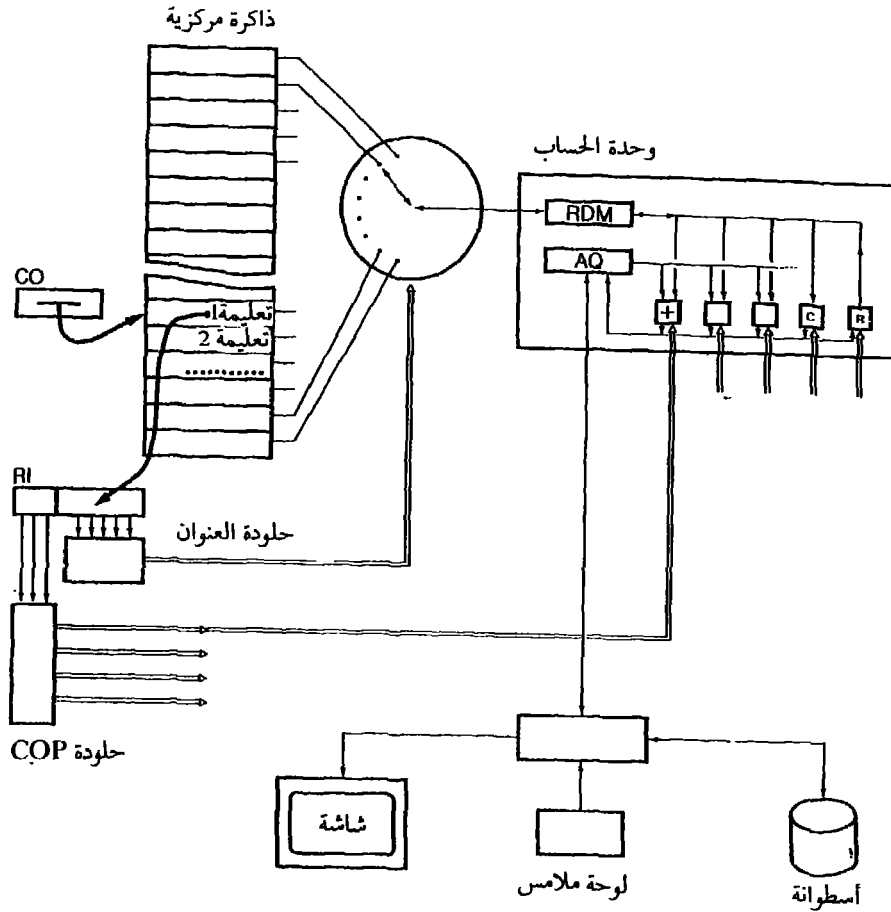
4.1 - خلاصة حول المكنة البسيطة

سنقوم بتوضيح الصيغ العملية للحاسب . إن جميع المكنات تستعمل هذه الأواليات الأساسية ، إضافة إلى بعض التعديلات التي سندرسها عند الحاجة . فلنحاول الآن أن نستخلص بعض الملاحظات .

ملاحظة 1

المكنة المشروحة أعلاه هي مكنة « بعنوان بسيط » ، أي أن التعليم الآلية لا تراجع سوى عنوان واحد وإذن متأثر واحد علني . في هذه الحالة ، لنفترض عدداً كبيراً من المؤثرات (operators) تستعمل متأثرين والنتيجة ، ذلك يعني أن أحد المتأثرين ثم النتيجة موجودان في المرمز . على بعض المكنات الأخرى قد نجد تعليمات تدعى « بعنوان مزدوج » .

(1) عندما تكون التعليمات ذات أطوال متغيرة (حالة الحاسبات IBM 360/370) يتقدم العداد CO بمقدار طول التعليم .



مخطط 5.1 - الحاسب ، المخطط العملياني

ملاحظة 2

لا تحتوي مكتنتنا سوى مركم واحد . هناك حاسبات أكثر فعالية يمكن أن تحتوي على عدد من المرافف التي تلعب دور المركم (هذه هي حالة المكنة IBM 360/370) . سيكون من الضروري أن نشير ، من داخل التعليمية ، إلى رقم المرفف الذي نعتمده كمركم .

ملاحظة 3

لنفترض ، كما في المخطط 3.1 ، أن ذاكرة المكنة تحتوي على 1000 خلية مرقمة من 0 إلى 999 . وهذا يعني أن :

1 - عداد البرنامج يحتوي على الأقل على ثلاثة مواقع عشرية تسمح له بمراجعة جميع عناوين الذاكرة المركزية ؛

2- ان حقل عنوان التعليم ، ولنفس السبب ، يجب أن يسمح بتسجيل الأعداد من 0 إلى 999 .

ملاحظة 4

بعض التعليقات يمكن أن لا تُراجع بواسطة عنوان ما . تظهر هذه الحالة ، مثلاً ، عندما لا نستعمل سوى AQ (عكس إشارة AQ ، تفسير AQ ، الإزاحة ، . . .) . ولكن من الممكن ، عند الحاجة ، إستعمال حقل العنوان لغايات أخرى . قد يحدث ، على بعض المكثات ، أن يكون حقل العنوان مستعملاً ككود لعملية ثانوية ، مما يؤدي إلى زيادة عدد التعليقات بدون تعديل لحجم الحقل COP . أما الكود الثانوي فيُميز التعليم الخاصة التي تنتمي إلى الفئة المحددة بواسطة الكود الرئيسي .

ملاحظة 5

الحجم (هنا يقاس بعدد المواقع العشرية) للحقل COP يُحدّد العدد الأقصى للدارات - أي للتعليقات الآلية - التي تراجع عنواناً وحيداً يمكن أن تحتويه وحدة الحساب .

5.1 . الحاسب ، العرض الكلاسيكي

بعد هذا المدخل ، نعود إلى عرض أكثر كلاسيكية للحاسب . لقد جرت العادة أن تُميز بين الأعضاء التالية :

الوحدة المركزية وتحتوي :

- الوحدة الجبرية والمنطقية (دارات عمليات ومرافق للحساب) ،

وحدة التحكم وتتألف من :

- مرافق التحكم ،

- عداد البرنامج ،

- الساعة .

الذاكرة المركزية وتتألف من خلايا (كلمات وبيانات) معنونة ،

- أدوات محيطية تسمح بالإدخال والإخراج في الذاكرة المركزية للمعلومات (برامج ومعطيات) المخزنة على نواقل خارجية

فلنذكر البعض منها :

- قارئ البطاقات ، والمثقبات ، والطابعات ،

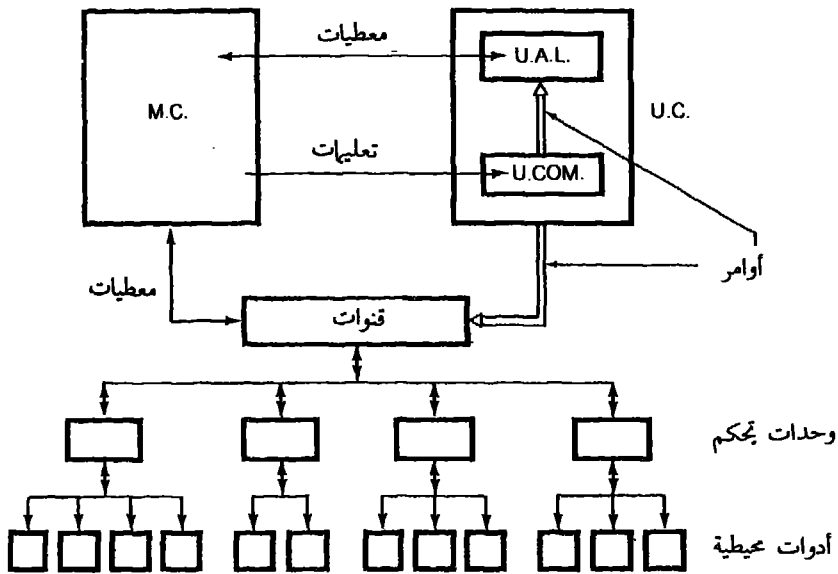
- بسّاطة الأشرطة ، الأسطوانات والطبول المغناطيسية ،

- لوحات ملامس ، - شاشات للعرض ،

- أدوات محيطية خاصة كراسم المنحنيات العاملة حسب النظام «off-line»

(الاشتغال المنزّل) .

- القنوات أو وحدات التبادل . وهي عبارة عن الأعضاء التي ، تحت قيادة الوحدة المركزية ، تؤمن بشكل لا تزامني إنتقال المعطيات من الذاكرة المركزية إلى الأدوات المحيطة . هذه الأولوية تسمح بتحرير موارد الوحدة المركزية خلال الوقت ، نسبياً « الطويل » ، للإدخال والإخراج (I/O)⁽¹⁾ . التزامن بين الوحدة المركزية والقنوات (Channels) يتأمن بواسطة نظام الانقطاع الذي سنتكلم عنه لاحقاً .
- وحدة المراقبة والتحكم (Control unit) وهي عبارة عن أجهزة وأدوات ، متكيفة مع كل نوع من المحيطات ، وتحقق عدداً من المهام الضرورية للإدخال والإخراج .



مخطط 6.1

(1) أعضاء الإدخال - الإخراج هي أجهزة الكتروميكانيكية تمثل إذن نوعاً من القصور . إن قراءة بطاقة معينة قد تطول نحو 100 ميللي ثانية في حين أن وقت تنفيذ تعليمة لا يدوم أكثر من الميكروثانية μs (10^{-6} ثانية)

2 تكويد المعلومات

الإستعمال الكثير للنظام العشري جعلنا معتادين عليه ، وهذا الإعتياد جعل البعض يخشى من إستعمال نظام آخر للترقيم . ولكن تكنولوجيا الحاسبات تفرض علينا دراسة أنظمة تكويد مختلفة . يجب أن نشير إلى أن التمثيل الثنائي للمعلومات في المكنة لا يحمل أي تعديل لصيغة العمل المشروحة في الفصل الأول ، وهذا من الأسباب التي جعلتنا لا نبدأ الكتاب بهذا الفصل ، راجين أن يكون عرضنا أكثر وضوحاً .

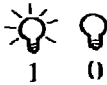
يتألف نظام التكويد من مجموعة قواعد التحويل التي تسمح بالعبور من تمثيل للمعلومات (نص فرنسي مثلاً) إلى ترميز آخر (نص بكود مورس . .) والعكس بالعكس .

الترميز الثنائي هو مفروض لأنه يسمح بتمثيل بسيط لمضمون الذاكرة والمراسف في الحاسب⁽¹⁾ . ويبدو أنه لترميز عدد n من حالات صمام كهربائي ، مولّع أو مطفأ ، فإن التمثيل الثنائي هو الأبسط باعتماد الاتفاق التالي :

1 - حالة « الضوء »

0 - حالة الإنطفاء

إذاً يرمز إلى الحالة بواسطة :



(1) دون الدخول في التفاصيل التكنولوجية، تمثل المعلومات داخل الآلة بواسطة عناصر تمتلك حالتين فيزيائيتين مختلفتين .

قد نلاحظ أن مجموعة من صمامين يمكن أن تكون موجودة في عدد $4=2^2$ من الحالات المختلفة التي نرمز إليها على الشكل التالي :

0	0	حالة «0»
0	1	حالة «1»
1	0	حالة «2»
1	1	حالة «3»

ولكن بإمكاننا تكويد :

الحالة «0» : الصمامان هما في حالة الإنطفاء
 الحالة «1» : الصمام اليسار هو مطفأ ، والصمام الأيمن مؤلّع ، الخ
 وبشكل عام ، فإن مجموعة من n من الصمامات يمكن أن تكون موجودة في 2^n حالة مختلفة . يجب تقريب ذلك من الفعل الذي يسمح بواسطة n رقم ثنائي بأن نعدّ من 0 إلى $2^n - 1$.

1.2 . أنظمة الترقيم :

لو افترضنا أن a_i تمثّل مجموعة الرموز المستعملة لتحديد عدد بالقاعدة B ، فإن العدد الحقيقي R يُكتب على الشكل التالي :

$$\frac{a_n a_{n-1} \dots a_1 a_0}{\text{القسم الصحيح}} , \frac{a_{-1} a_{-2} \dots}{\text{القسم العشري}}$$

وقيمته هي :

$$R = \frac{a_n B^n + a_{n-1} B^{n-1} + \dots + a_0 B^0}{\text{القسم الصحيح}} + \frac{a_{-1} B^{-1} + a_{-2} B^{-2} + \dots}{\text{القسم العشري}}$$

وفي النظام العشري فإن المجموعة a_i تتألف من الرموز :

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9$$

وفي الثنائي : 0 و 1 .

وفي النظام الثماني : 0, 1, 2, 3, 4, 5, 6, 7 .

وفي النظام السادس عشري (16) : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

$$F, E, D, C, B, A$$

إنَّ أسَّات القاعدة B^0 ، B^1 ، B^2 ، ... ، B^{-1} ، ... تدعى أوزان الأرقام .
الجدول 1.2 يعطي قيم بعض الأوزان بالنظام العشري :

القاعدة	B^3	B^2	B^1	B^0	B^{-1}	B^{-2}
10	1000	100	10	1	0,1	0,01
2	8	4	2	1	0,5	0,25
8	512	64	8	1	0,125	0,015625
16	4096	256	16	1	0,0625	0,00390625

جدول 1.2

هكذا فالعدد 13 في القاعدة 10 يعادل $(3.10^0 + 1.10^1)$
ويُكتب على الشكل التالي : $(1.2^0 + 0.2^1 + 1.2^2 + 1.2^3)$ في النظام الثنائي .

15 في النظام الثنائي : $1.8^1 + 5.8^0$
D في النظام السادس عشري : $(13.16^0 \text{ أي } D.16^0)$
والعدد 0,75 في النظام العشري : $(5.10^{-2} + 7.10^{-1})$
يكتب : 0,11 في النظام الثنائي : $(1.2^{-2} + 1.2^{-1})$
0,6 في النظام الثنائي : (6.8^{-1}) .
و C, 0 في النظام السادس عشري : $C.16^{-1} \text{ أي } 12.16^{-1}$.

وفي المكنة ، تُمثَّل الأعداد بشكل مكوَّن ثنائياً . ويمكن أن يحتاج عدد عشري كسري إلى سلسلة طويلة ، أو لا نهائية ، من 0 و 1 . وبما أن الذاكرة والمراصف لها أبعاد محدَّدة عند تصميم المكنة ، لذا ، فقد يحدث تحويل عشري / ثنائي عند الحساب ، أو قد يحدث بتر لقسم من المعلومات مما يؤدي إلى فقدان الدقة في الحساب . وهذه من المشاكل التي يجب الانتباه إليها ولذا من الواجب القيام بعدد كبير من الحسابات التكرارية .

من المهم أن نلاحظ ، أنه عند إزاحة الفاصلة r موقع لجهة اليسار أو لجهة اليمين فإن هذا يؤدي إلى ضرب العدد أو قسمته على 10^r مثلاً : 13,75 يمثل بواسطة العدد 1101,11 في النظام الثنائي ، ولكن 11011,1 يعادل 27,5 و 110,111 يعادل 6,875 .

عشري	ثنائي	سادس عشري	ثماني
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

جدول 2.2

2.2 . تغيير القاعدة

سنترك للقارئ أن يعود للمراجع إذا رغب بذلك . وسنذكر ، بواسطة بعض الأمثلة ، إن التحويلات الثنائية / الثمانية والثنائية / السادس عشرية هي متزامنة لأن القواعد 8 و16 هي عبارة عن أسس صحيحة للقاعدة 2 .

ينقلب العدد الثنائي إلى سادس عشري بدءاً من كل جهة من موقع الفاصلة . وبتقطيع العدد إلى أقسام مؤلفة من أربعة أرقام ثنائية أو بتات⁽¹⁾ وبتأويل كل قسم :

$$\begin{array}{ccccccc} \text{ثنائي} & & & & & & \\ \hline 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & , & 1 & 1 & 0 & 0 & 1 \\ \hline 2 & & 6 & & B & & C & & 8 & & & & & & \\ \text{سادس عشري} & & & & & & & & & & & & & & \end{array}$$

الرقم الأخير «8» نحصل عليه بتوسيع الرقم 1 بوضع أصفار لجهة اليمين . التحويل الثنائي / الثماني يتم بتقطيع العدد الثنائي إلى أقسام مؤلفة من ثلاثة أرقام . نحصل عندها على 1153, 62 في النظام الثماني . التحويل المعاكس هو بدسهي .

3.2 . الفائدة من النظامين السادس عشري والثنائي

سنرى أن كل كلمة آلية هي مكوّنة من عدد متحول ، يتعلّق بالحاسب ، من العناصر التي تدعى بتات⁽¹⁾ (bit) . كل عنصر يمكن أن يكون موجوداً ، كما هي الحالة

(1) من BIT وهو اختصار للمصطلح الأميركي Binary digiT ، أي رقم ثنائي .

بالنسبة للصَّمَام ، في واحدة من حالتين فيزيائيتين ، لذا يصبح من الطبيعي ترميز حالة البتة بواسطة 0 أو 1 ومضمون الكلمة - الآلية ، ليس كما في الفصل الأول بواسطة رقم عشري ، بل بواسطة سلسلة من الأرقام 0 أو 1 ، ويمكن تفسير مجموعة البتات كعدد تمثّل في النظام الثنائي .

الأحجام ، المحددة بعدد البتات ، للكلمات - الآلية التي نلتقيها عادة في الحاسبات هي بطول 8 (الميكروبروسسور) ، 16 ، 24 ، 32 (IBM 360/370) ، 36 ، 48 و 60 بتة . عند تمثيل مضمون كلمة - ذاكرة على ورقة فهذا يتطلب من 16 إلى 60 رمزاً . التمثيل السادس عشري والثاني يظهران إذن مفيدتين مهمين كثيراً لأنها يُقسَّمان على 4 أو على 3 عدد الرموز المطلوب كتابتها وذلك مع المحافظة على إمكانية تحويلها فوراً إلى النظام الثنائي . ولكن النسخ اليدوي لعدد محدّد بالنظام السادس عشري هو منبع لعدد أقل من الأخطاء منه في حال كتابته في النظام الثنائي . لذلك فللقارئ فائدة من الإعتياد على هذا النوع من التمثيل المعتمد لتمثيل المعلومات في الذاكرة .

4.2 . الحساب في النظامين الثنائي والسادس عشري
لن نقوم سوى بإعطاء بعض الأمثلة التي يجب أن تسمح للقارئ بإجراء بعض العمليات البسيطة بالجمع والطرح .

في النظام الثنائي :

$$\begin{array}{r} 0 \quad 0 \quad 1 \quad 1 \\ +0 \quad +1 \quad +0 \quad +1 \\ 0 \quad 1 \quad 1 \quad 10 \end{array}$$

$$\begin{array}{r} 1101 \\ +111 \\ 10100 \end{array}$$

مثلاً :

$$\begin{array}{r} D \\ +7 \\ 14 \end{array}$$

في النظام السادس عشري :

في النظام السادس عشري من العملي تحويل كل رقم إلى النظام العشري ، وإجراء العملية في هذا النظام ومن ثم تحويل النتيجة . مثلاً :
 $D_{16}=13_{10}$, $7_{16}=7_{10}$, $13+7=20_{10}=16+4$
نضع 4 ونحفظ باليد 1 ، إلخ :

$$\begin{array}{r} 3F2 \\ +1A4 \\ \hline 596 \end{array} \quad \begin{array}{r} 3F2 \\ -1A4 \\ \hline 24E \end{array}$$

بنفس الطريقة نقوم بإجراء الطرح $4 - 2_{16}$ تصبح $4 - 16 + 2$ أي E وباليه 1 . . .
حسب نفس الصيغة سنستطيع إجراء الحساب في النظام الثنائي . وباستطاعة
القارئ أن يتمرّن بوجود الأمثلة المعطاة في نهاية الفصل .

5.2 . التمثيل الداخلي للمعطيات

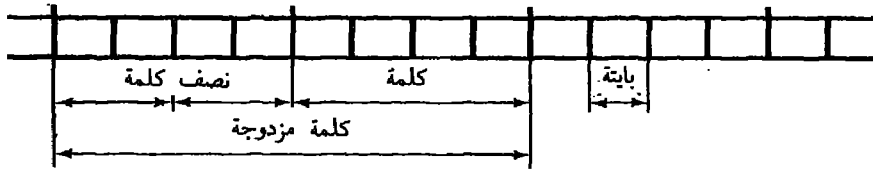
1.5.2 . الذاكرة

حتى هذا الوقت إعتبرنا إن الذاكرة هي مؤلفة من خلايا مرقّمة بدءاً من 0 ،
الخلية هي الكلمة - الآلية والعناوين هي عناوين الكلمات .

سنقوم بتحديد الأشياء . المكثات IBM 360/370 تتمتع بكلمة - آلية من 32 بتة
مرقّمة من اليسار إلى اليمين من 0 إلى 31 . تُقسّم الكلمة إلى أربع بايتات (تشكيلة من
8 بتات) . والبايتة هي قابلة للعنونة . ستتكلّم عن الذاكرة المعنونة بالسّيات (وسنرى
إن ألسمة قابلة للتمثيل بواسطة 8 بتات) مقابلة مع بعض المكثات حيث الذاكرة معنونة
بالكلمات . عنوان الكلمة هو إذاً عنوان البايّة الأولى من الكلمة . في النهاية نوجز ما
يلي :

- جهات النصف كلمات هي بعناوين مزدوجة ؛
- جهات الكلمات هي بعناوين قابلة للقسمّة على أربعة ؛
- جهات الكلمات المزدوجة تتمتع بعناوين قابلة للقسمّة على 8 ؛

ومع إن الذاكرة هي قابلة للعنونة في مستوى البايّة ، يجب السهر على المحافظة
على هذا التقسيم للمعطيات الممثّلة بواسطة نصف كلمة ، كلمة ، أو كلمة مزدوجة .



شكل 3.2

2.5.2 . تمثيل المعطيات الارقمية

بإمكاننا تكويد نوعين من المعلومات في الذاكرة : المعطيات الرقمية والتي هي
عبارة عن تشكيلات ثنائية مرتبطة بمعنى رقمي ، والمعطيات من نوع سيات والمعالجة
كوحداث غير رقمية .

لقد كان من الملائم عند تصوّر مكنات IBM 360/370 ، تكويد السهات بواسطة 8 بتات . هذا النظام يسمح بتكويد 2^8 ، أي ما مجموعه 256 كوداً مختلفاً . هذا التصوّر هو واسع الإنتشار ، ولكن هناك مكنات أخرى تستعمل تكويد السهات بواسطة 6 بتات -تُحدّد مجموعة السهات المتوفّرة بالعدد 64 سمة .

قد يبدو لنا مفاجئاً إعتداد كود لتمثيل السهات بواسطة 8 بتات . فلنلاحظ ببساطة إن هذا النظام يسمح لنا بالحصول على ألفباء واسعة تحتوي على السهات الكبيرة ، والصغيرة ، والسهات العشر العشرية وبعض السهات الخاصة ، كإشارات العمليات ، وعلامات الوقف ، والفسحة ، الخ .

الكود الداخلي لتمثيل السهات ، والمستعمل على المكنات IBM 360/370 هو EBCDIC (Extended Binary Coded Decimal Interchange Code) . يُرمز إلى الحرف A بواسطة الكود 11000001 ، أي C1 بالترميز السادس عشري . ويكوّد الحرف «B» بواسطة C2 وهكذا دواليك . لائحة الأكواد موجودة في الملحق . مثلاً : لنفترض إن مضمون حيّز الذاكرة هو التالي :

0	0	C	1	E	2	E	2	C	5	D	4	C	2	D	3	C	5	E	4	D	9	4	0	F	3	F	7	F	0	0	0
100								104								108								112							

تأويل هذه السلسلة من 14 بايتة ، والتي تبدأ بالعنوان 100 ، هو حسب الكود «ASSEMBLER 370» .

نشر إلى وجود علاقة تراتبية بين القيم الثنائية المستعملة للتكويد :

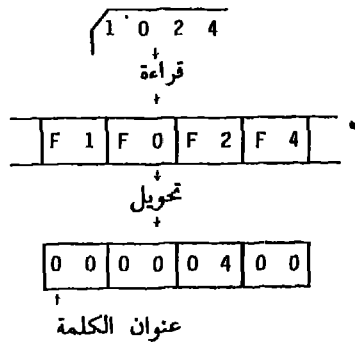
$$40 < C1 < C2 < \dots < F0 < F1 < \dots < F9$$

وهذا يمكن أن يترجم بواسطة :

كود الأرقام > ... > كود B > كود A > كود القسمة .

هذه الخصوصية هي مستعملة للترتيب الأبجدي .

يجب أن تُميّز بين التمثيل الأبجدي والتمثيل الرقمي . المثل التالي يوضح لنا التحويل المعتمد لمعطى مقروء من البطاقة ومحوّل إلى ثنائي .



ذاكرة (حيّز إدخال)

تمثيل بطريقة السهات

ذاكرة (متحوّلة مؤشّرة في)

FORTRAN في لائحة الأمر READ .

تمثيل بطريقة الفاصلة الثابتة

التمثيل السهائي يُقال عنه أيضاً « القابل للتنقيح » لأنه ضمن هذا الشكل يجب أن تكون المعلومات موجودة قبل أن تستلمها الطابعة لطبعها .

3.5.2 . تمثيل المعطيات الرقمية

المعتادون على لغة فورتران يعلمون أن المتحولة أو الثابتة يجب أن تُمثّل دائماً في المكنة بواسطة كلمة (أو كلمة مزدوجة عندما يكون الحيزُ مصرّحاً عنه بدقة مزدوجة) . ويعلمون أيضاً إن هذه اللغة تستعمل نوعين أساسيين من التمثيل الداخلي للمعطيات الرقمية : النوع الصحيح (integer) والنوع العائم (real) .

أما المعتادون على لغة كوبرول فلا يجهلون ان الحسابات الجارية بهذه اللغة تتم بواسطة تمثيل مجهول من لغة فورتران : التمثيل العشري المتراص . سنجد هذه الطرق الأربع في تكويد الأعداد في مستوى المكنة : الطريقة « الفاصلة الثابتة (fixed point) » (صحيح بلغة فورتران) ، والعائم البسيط والعائم الموسّع والصيغة العشرية المترابطة . نشير إلى أنّ مع كلّ نوع من هذه التمثيلات تتلاءم مجموعة من المؤثرات (دارات الكترونية ، + ، - ، ...) ، صالحة للعمل بهذه التشكيلات الثنائية . وفي النتيجة فإن المكنات تحتوي على أربع مجموعات من التعليمات الجبرية .

أ - التمثيل بفاصلة ثابتة

بهذه التسمية يجب أن نفهم « فاصلة ثابتة إتفاقياً » . هكذا ، فالفاصلة ، عنصر أساسي من قيمة العدد ، لا تظهر أبداً في التمثيل الداخلي للعدد في الذاكرة . ولقد لاحظنا (في الفقرة 1.2) إن التشكيلات الثنائية المعتمدة لـ n ، $2n$ و $n/2$ لا تختلف إلا بواسطة موقع الفاصلة ، لذا ، فإن 1001 يمكن أن تُمثّل القيمة 9 إذا إعتبرنا إن الفاصلة موجودة لجهة اليمين ، أو 0,5625 إذا إعتبرنا إن الفاصلة موجودة في أقصى اليسار النظام IBM 360/370 يفترض الفاصلة موضوعة لجهة اليمين . وللتأكد من ذلك يكفي ملاحظة التعليمات التي تسمح بجمع المعطيات بطول مختلف (كلمة أو نصف كلمة) . إن عملية التسطير للمعلومات تتم لجهة اليمين . هذا التمثيل هو إذاً التمثيل الصحيح . وهناك بعض المصممين الآخرين الذين إعتمدوا الإتفاق المعاكس ، أي الفاصلة لجهة اليسار .

تُكوّد الأعداد حسب النظام الثنائي في كلمة - آلية . البتة ذات الوزن الأكبر (البتة الموجودة لجهة اليسار) ترمز إلى الإشارة الحسابية . إذا كانت تساوي 0 ، يكون العدد إيجابياً ، أما إذا كانت تعادل 1 فمعنى ذلك أن العدد هو سلبى .

بواسطة n بتة باستطاعتنا تعداد من 0 حتى $2^n - 1$. وإذا حجزنا بتة للإشارة فيكون بإمكاننا تمثيل الأعداد الصحيحة I بحيث إن :

$$-2^{n-1} \leq I \leq 2^{n-1} - 1$$

إذا كانت $n = 16$: $- 32768 \leq I \leq + 32767$ -

تمثيل الأعداد الإيجابية لا يفترض أية مشكلة ، والتأويل العشري نحصل عليه بضرب كل بته بالوزن المعتمد للموقع . وفي المقابل يجب أن نعتد إتفاقاً جديداً للأعداد السلبية .

تمثيل الإشارة والقيمة المطلقة

الفكرة التي تخطر لنا تقوم على إعتبار البته ذات الوزن الأقوى ترمز إلى الإشارة والباقي يرمز إلى القيمة المطلقة للعدد . حسب هذا الإتفاق ، المثل بأربع بتات :

$$\begin{array}{r} 0101 \\ 1101 \\ \hline 10010 \end{array} \begin{array}{l} + 5 \text{ يكتب :} \\ - 5 \text{ يكتب :} \\ \text{نتيجة الجمع :} \end{array}$$

هذه النتيجة هي ليست حقيقية .

هذا التمثيل يُجتم علينا إذاً ، للحصول على النتيجة الصحيحة ، أن نفحص الإشارات المرتبطة بالتأثرات قبل إجراء العمليات . لا يجب معالجة الأعداد السلبية والإيجابية بنفس الطريقة . يمكن للقارئ أن يقتنع بأن إعتد هذه الصيغة يحتم علينا إعتد منطق ألكتروني أكثر تعقيداً . وقد جرى التخلي عنه اليوم .

التمثيل المدعو مُكْمَل 1 (1 Complement)

عكس العدد (أو ضده) . نحصل عليه بأخذ عكس كل بته . بما فيها بته الإشارة . هكذا :

$$\begin{array}{r} 0101 \\ 1010 \\ \hline 1111 \end{array} \begin{array}{l} + 5 \text{ تكتب :} \\ - 5 \text{ تكتب :} \end{array}$$

وبنتيجة الجمع نحصل على

أي ، أن مُكْمَل 1 هو 0000

من الممكن إعتبار إن هذا النوع من التمثيل يؤدي إلى إدخال 0 إيجابي وصفر سلمي . المسائل المطروحة في نهاية الفصل تشرح سيئات هذه الاتفاقات وفوائد الاتفاقات اللاحقة

التمثيل المدعو « مُكْمَل إلى 2 » (Two complement)

هو التمثيل المعتمد على المكنات IBM 360/370 . يُثّل كل عدد سلمي بواسطة المُكْمَل إلى 2^n لعكس العدد . ولو إفترضنا إن X هو العدد ، وأن \bar{X} هو مُكْمَل العدد X إلى 2^n ، نحصل إذاً على العلاقة التالية $X + \bar{X} = 2^n$. الإتفاق حول الإشارة هو كالسابق . ونشير إلى أن المعطيات الرقمية هي مكوّدة بأطوال ثابتة ، هي الكلمات -

الآلية . وللمكنات IBM 360/370 ، n تعادل 32 . ولتسهيل العمل ، فإننا سنعالج مسائل تعمل بأربع أو ثمان بتات .

وبالتكوين بواسطة أربع بتات ، حيث البتة اليسرى هي بتة الإشارة ، فإن كود العدد -5 هو المعادل الثنائي لـ $2^4 - 5 = 11$ إذن :

$$\begin{array}{r} +5 \\ -5 \\ \hline 0 \end{array} \quad \begin{array}{r} 0101 \\ 1011 \\ \hline 10000 \end{array}$$

وبإهمال الحاصل بعد موقع الإشارة نحصل على صفر .
الطريقة للحصول على المكمل إلى 2 لعدد ما تكمن بتكملة العدد إلى 1 وبعد ذلك إضافة 1 إليه . تتم العمليات على جميع البتات بما فيها بتة الإشارة .
مثلاً :

$$\begin{array}{r} +5 \\ 1 \text{ إلى مكمل} \\ +1 \\ \hline 0101 \\ 1010 \\ +1 \\ \hline 1011 \end{array}$$

فلنلاحظ إنه إذا كنا نعمل على عدد ثنائي مُمثل بالترقيم السادس عشري ، فإن المكمل إلى 2^n يصبح مُكملاً إلى 16^n سنحصل على التمثيل السادس عشري للعدد المعكوس بتكملة كل رقم إلى F وإضافة 1 .
مثلاً : على ثمان بتات :

$$\begin{array}{r} 0100 \ 1101 \rightarrow 4D \\ \downarrow \\ B2 \\ +1 \\ \hline 1011 \ 0011 \leftarrow B3 \end{array}$$

انتقال العدد ، المُمثل بواسطة 16 بتة ، في مرصف بطول 32 بتة سيتم بواسطة انتقال بسيط إلى اليسار للبتة ذات الوزن الأكبر :

$$\begin{array}{r} 0A1C \rightarrow 0000 \ 0A1C \\ B0D3 \rightarrow FFFF \ B0D3 \end{array}$$

حالة الفيض عن السعة (Over flow) ، يمكن أن تحدث عند إجراء عملية معينة وذلك عندما يكون كلا المتأثرين بنفس الإشارة والنتيجة تصبح بإشارة معاكسة .
لنعت بعض الأمثلة على معطيات ممثلة بواسطة أربع بتات . مجموعة الأعداد القابلة للتمثيل هي :

1111	-1	0000	0
1110	-2	0001	+1
1101	-3	0010	+2
1100	-4	0011	+3
1011	-5	0100	+4
1010	-6	0101	+5
1001	-7	0110	+6
1000	-8	0111	+7

بالإمكان إهمال المرحّل اليسار بدءاً من موقع الإشارة ، إذا كان كلا المتأثرين بنفس الإشارة ، والنتيجة بنفس الإشارة .

وجود المرحّل ، ويُدعى (Carry) في المصطلحات الأنكلوسكسونية ، ليس هو إشارة خطأ في الحساب .

سنلاحظ في النهاية إن العدد الأصغر القابل للتمثيل هو 2^{-1} -2- والأكبر هو 2^{-1} -1- وإن الطرح يمكن أن يتم بواسطة الجمع إلى مكمل 2 .

نتيجة		مرحلّ مفقود	عشري
خطأ DDC (1)	$\begin{array}{r} 0\ 111 \\ 0\ 111 \\ \hline 1\ 110 \end{array}$		$\begin{array}{r} +7 \\ +7 \\ \hline 14 \end{array}$
خطأ DDC (1)	$\begin{array}{r} 0\ 100 \\ 0\ 101 \\ \hline 1\ 001 \end{array}$		$\begin{array}{r} +4 \\ +5 \\ \hline 9 \end{array}$
صحيح	$\begin{array}{r} 0\ 100 \\ 1\ 011 \\ \hline 1\ 111 \end{array}$		$\begin{array}{r} +4 \\ -5 \\ \hline -1 \end{array}$
خطأ DDC (1)	$\begin{array}{r} 1\ 100 \\ 1\ 011 \\ \hline 0\ 111 \end{array}$	1	$\begin{array}{r} -4 \\ -5 \\ \hline -9 \end{array}$
صحيح	$\begin{array}{r} 1\ 101 \\ 0\ 011 \\ \hline 0\ 000 \end{array}$	1	$\begin{array}{r} -3 \\ +3 \\ \hline 0 \end{array}$

DDC (1) : الفيض عن السعة (over flow)

ب - التمثيل بفاصلة متحركة

الحساب العلمي يستعمل عادة أعداداً بأحجام كبيرة جداً أو صغيرة جداً ولكن ممثلة بواسطة عدد مُحدّد من الأرقام . النوع فاصلة ثابتة لا يسمح بالتمثيل البسيط لهذه الأعداد ، ولذلك إعتدنا طريقة أخرى في التكويد المركّب من قسمين :

- المُميّزة (الأسّ المعيّن) التي تعطي الحجم .
- القسم العشري (mantisse) الذي يحدد الأرقام ذات الأوزان الكبرى .

هكذا فيإمكاننا تحديد العدد على الشكل التالي :

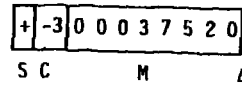
$$S.M.B^c$$

حيث S هي الاشارة ، M القسم العشري (mantisse) ، و B عدد ثابت (2 ، 10 ، أو 16 حسب المكتة) ، C هي الأسّ المعيّن .

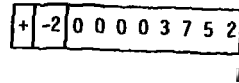
كما في الفاصلة الثابتة ، فإن الفاصلة لا تظهر في التكويد الداخلي ولكنها توضع عادة إلى يمين أو إلى يسار القسم العشري M . هكذا ، فلنظام بقاعدة $B=10$ ، يُكتب العدد 37,52 على الشكل التالي⁽¹⁾ :

1 - الفاصلة لجهة يمين القسم العشري .

$$(1) \quad 37520.10^{-3}$$

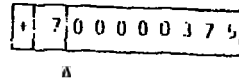


$$(2) \quad 3752.10^{-2}$$

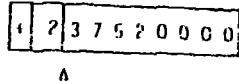


2 - الفاصلة إلى يسار القسم العشري ،

$$(3) \quad 0,000003752.10^7$$



$$(4) \quad 0,3752.10^2$$



نلاحظ ، في الحالة التي تكون فيها الفاصلة موجودة إلى يسار القسم العشري ، إن التمثيل (4) يعطي عدداً أكبر من الأرقام ذات المعنى (Significants digits) من التمثيل

(1) Δ : رمز يشير إلى موقع الفاصلة .

(3) ، في الحالة التي يكون فيها عدد الأرقام المحجوزة للقسم العشري ثابتاً . التمثيل (4) يُدعى موحد التنظيم المعايير (normalized) . وهو يتناسب مع حصر الأرقام ذات المعنى من القسم العشري لجهة اليسار . هذا التمثيل يسمح بأكثر دقة ممكنة .
من الممكن أن نعبر عن تمثيل معين إلى تمثيل معايير آخر بواسطة إزاحة الأرقام وتعديل الأس .

إذا كانت $B = 10$ ، فإن الإزاحة إلى اليسار لموقع رقم يؤدي إلى تنقيص الأس المعين 1 .

أما إذا كانت $B = 16$ ، فإن الإزاحة إلى اليسار لرقم سادس عشري من القسم العشري سيؤدي إلى تنقيص 1 من الأس المعين . وهكذا سيكون العدد ممثلاً بشكل معايير عندما لا يكون الرقم السادس عشري ذو الوزن الأكبر من القسم العشري صفراً . سنشير إلى أن الإزاحة لموقع سادس عشري يُترجم إزاحة أربعة مواقع ثنائية . على الحاسبات IBM 360/370 :

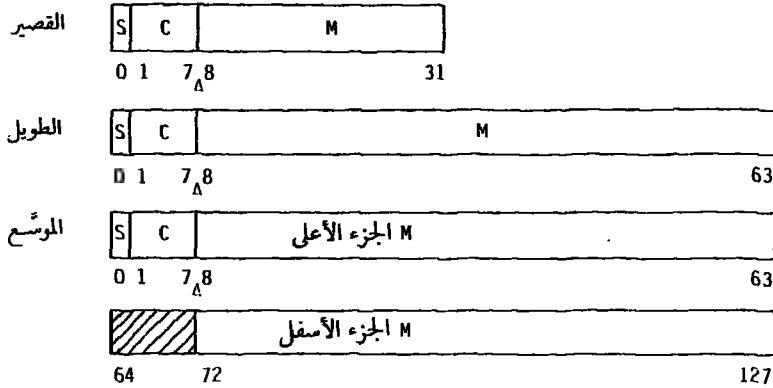
- الإشارة S من العدد هي مكودة على بته واحدة ($0 = +$ ، $1 = -$) ؛
- القاعدة B هي 16 ؛
- يفترض أن تكون الفاصلة إلى يسار القسم العشري الذي يُمثل عدداً أصغر من 1 .
- العدد الثنائي المكود في الحيز C بطول 7 بتات والمحفوظ للأس المعين ، لا يُمثل أبداً قيمة الأس المعين E لـ 16 ولكن :

$$C = 64_{10} + E$$

لذا فهناك مشكلة في تكويد إشارة الأس كي نحصل على قوى سلبية وإيجابية للقاعدة B بدلاً من اعتماد ترميز شبيه بالمكمل إلى 2 ، لقد جرى إختيار اعتماد القوة صفر في التكويد المناسب للقيمة الوسطية للأعداد القصوى 0 و $2^7 - 1$ أي 64_{10} أو 40_{16} أو 1000000_2 . هكذا ، عندما تكون $C = 64_{10}$ فإن قيمة العدد هي $S.16^0.M$ ، وعندما تكون $C > 64$ فإن قيمة العدد هي $C.S.16^E.M$ متغيرة من 0 إلى 127 وبالتالي $-64 \leq E \leq +63$.

للحصول على E يكفي ، في النظام السادس عشري ، أن نطرح 40_{16} : 46_{16} تناسب $E=6$ و $3F_{16}$ تناسب $E = -1$.

يوجد على الحاسبات IBM 370 ثلاثة أشكال بفاصلة متحركة . الأعداد بالفاصلة المتحركة الصغيرة تحتل كلمة - آلية ، والأعداد الطويلة تحتل كلمتين - آيتين ، والأعداد الموسعة تشغل أربع كلمات . الشكل الأخير هو غير موجود على المكنات 360 .



4.2 خطط

الأشكال الثلاثة تسمح بتكويد أعداد بنفس الحجم . وتختلف بواسطة عدد الأرقام ذات المعنى التي تقدّمها . العدد P هو :
بالشكل القصير :

$$16^{-65} \leq P \leq (1-16^{-6}).16^{63}$$

7 أرقام عشرية ذات معنى

$$16^{-65} \leq P \leq (1-16^{-14}).16^{63}$$

- بالشكل الطويل

16 رقماً عشرياً ذا معنى .

$$16^{-65} \leq P \leq (1-16^{-28}).16^{63}$$

- بالشكل الموسّع

33 رقماً عشرياً ذا معنى .

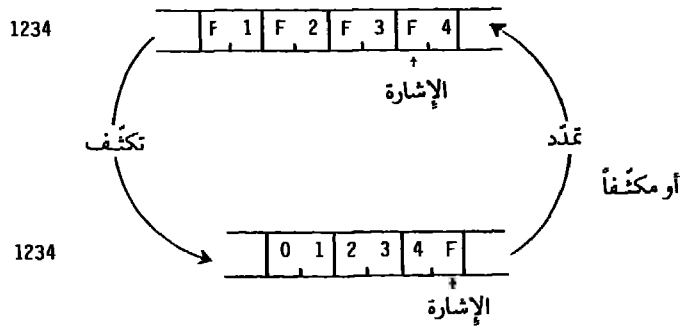
وفي الحالات الثلاث يكون معنا تقريباً : $5,4.10^{-79} \leq P \leq 7,2.10^{75}$

أما الحسابات بواسطة هذه الطرق في التمثيل فقد تؤدي إلى فيض عن السعة (Overflow) عندما نحصل على قيم كبيرة جداً أو صغيرة وتدعى Overflow أو Underflow للأعداد بالفاصلة المتحركة .

مثلاً : التمثيل بفاصلة متحركة

C 2 1 9 0 0 0 0	-25	$-(1.16^{-1} + 9.16^{-2}) 16^2$
C 1 1 0 0 0 0 0	-1	$-(1.16^{-1}) 16^1$
0 0 0 0 0 0 0 0	0	$0 . 16^{-64}$

ج - التمثيل العشري
يمكن أن يتم تمثيل العدد بواسطة النظام العشري المكوّن ثنائياً (DCB) موسّعاً ،
أي على شكل سمات .



هذا التمثيل ، الواسع الانتشار في الإدارة ، هو أقل « تراصاً » من سوابقه . لا يوجد أي طول ضمني لها : توضع المعطيات بداخل بايتات . سنرى ان التعليقات PACK و UNPK تسمح بالعبور من شكل إلى آخر .
الفاصلة ، كما رأينا ، ليست ممثلة . وإن تنظيم موقعها والاصطفاف المحتمل المناسب يقع على عاتق المبرمج . ونشير إلى المواقع المختلفة للإشارة . القيم السادس عشرية A, C, F, E يجري تأويلها وكأنها « + » . أما B و D فيؤوّلان وكأنها « - » .

تمارين

تمرين 1.2 - غيّر إلى النظام الثنائي والسادس عشري ، الأعداد العشرية التالية :
15 35 256 1024 348,5 .

تمرين 2.2 - غيّر إلى النظام العشري والثنائي الأعداد السادس عشرية التالية :
3A FFF 1A3B ABC

تمرين 3.2 - إخصب المكمل إلى 2 للعدد 1A3B . أطرح 1A3B من العدد 2ABC .
أعطِ التمثيل الموسّع إلى 32 بتة للعدد 1A3B وكذلك لمكمله إلى 2 .

تمرين 4.2 - أعطِ القيم الرقمية العشرية التي نقوم بتأويلها :

C1F00000

- كمعطى ممثّل بفاصلة ثابتة حسب تكويد الاشارة والقيمة المطلقة .

- كعدد ممثّل بالمكمل إلى 2 .

- كعدد بفاصلة متحركة بطول قصير (هل هو معاير في هذه الحالة؟) .
هل بالإمكان اعتبار هذا التشكيل الثنائي كمعطى مكود بالشكل العشري ؟

ما هو نقيض (أو ضدّ) هذا العدد في كلّ من التمثيلات المذكورة ؟

تمرين 5.2 - عاير العدد بفاصلة متحركة C5032000 .

3 . العنوان المطلقة ، العنوان النسبية

1.3 . عموميات

في الفصل الأول عرضنا التعليمات - الآلية وكأنها مشكّلة من حقلين : الحقل كود العملية (operation code) وحقل العنوان . تحتوي التعليمة إذاً على العنوان المطلق للمتأثر ، أي عنوانه الفعلي أو الحقيقي بالنسبة للعنوان 0 من الذاكرة . هكذا فبرنامج جمع مضمون الجلايا 0 و1 وخزن النتيجة في العنوان 2 كان قد كتب على الشكل التالي:-

0		المتأثر الأول
1		المتأثر الثاني
2		النتيجة
3	8 8 0	$(0) + AQ$
4	9 0 1	$AQ+1 + AQ$
5	8 0 2	$AQ + (2)$

فلنفترض بأننا زرعنا هذا البرنامج (مجموعة مناطق العمل والتعليمات) ليس على العنوان 0 ولكن على العنوان 100 . سنكتب عند ذلك :

100		المتأثر الأول
101		المتأثر الثاني
102		النتيجة
103	8 8 1 0 0	$(100) + AQ$
104	9 0 1 0 1	$AQ+(101) + AQ$
105	8 0 1 0 2	$AQ + (102)$

نلاحظ أن كود العمليات لا يتغير ولكن العناوين قد جرى نقلها 100 موقع لأن التعليمات تعود إلى العناوين المطلقة . أو بشكل آخر ، فإن كتابة البرنامج تتعلق بالعنوان الفعلي لمكان البرنامج . هذا الإلزام ، الذي سنعرض سيئاته ، قد أجبر مُصممي المكونات على تعريف أوالية العنونة النسبية : حقل العنوان من التعليمات لا يعود إلى العنوان المطلق للمتأثر ولكن إلى عنوان نسبي حسب عنوان أساسي (مطلق) . وبالإجمال فإن حقل العنوان يعطي « المسافة » إلى موقع المتأثر بالنسبة إلى عنوان يُعتبر وكأنه أساس أو قاعدة (Base adresse) ويعرّف في لحظة زرع البرنامج في الذاكرة . العنوان الفعلي (المطلق) للمتأثر سيحسب ، في لحظة تنفيذ التعليمات ، بواسطة جمع العنوان المرجعي (الأساسي) إلى قيمة الإزاحة المحددة في التعليمات . سنعمد في ما يلي إلى شرح أواليات عدّة للعنونة تتواجد في نفس الوقت على الآلات الحالية .

2.3 . العنونة القاعدية

هي عنونة نسبية حيث المبدأ هو كما ورد أعلاه . يحتوي الحاسب على عدد من المراسف التي يمكن أن تستعمل كمراسف أساسية (قاعدية) ، ويجب على المبرمج :
- أن يختار المراسف الأساسي بواسطة أمر خاص .
- أن يُخزّن قيمة معينة في هذا المراسف ، قيمة ستكون عبارة عن العنوان الأساسي .
- كتابة البرنامج (معطيات وتعليمات) نسبة إلى عنوان معين يعادل عادة الصفر .
وفي لحظة التنفيذ يُسحق البرنامج في الذاكرة ، ويُخزّن قيمة العنوان القاعدي في المراسف القاعدي . عند تنفيذ كل تعليمات فإن العنوان الموجود في التعليمات (الإزاحة (déplacement) يُضاف أوتوماتيكياً إلى مضمون المراسف القاعدي للحصول على العنوان الفعلي للمتأثر .

ذاكرة

150		المتأثر الأول
151		المتأثر الثاني
152		النتيجة
153	8 8 0	
154	9 0 1	
155	8 0 2	
156		

يُكتب البرنامج دون الإهتمام بالعنوان الفعلي لمكان خزن البرنامج . ونُحسب جميع العناوين نسبة إلى العنوان صفر (بداية البرنامج) .

ولنفترض إن بداية البرنامج (العنوان النسبي صفر) موجودة على العنوان الفعلي 150 ، وهي قيمة سيتم تخزينها في مصرف القاعدة⁽¹⁾ . إذا فالعنوان النسبي n للبرنامج يناسب العنوان الفعلي $n + 150$. . . والبرنامج سيقوم بتنفيذ العملية :
 $(152) \rightarrow (151) + (150)$

لدينا إذن العلاقة التالية :

العنوان الفعلي = العنوان القاعدي + العنوان الموجود في التعليمات

نشير إلى أن عملية الجمع تتم ديناميكياً ، في لحظة تنفيذ كل تعليمات . يبدو من البديهي أن المبرمج لا يجب أن يُعدل مضمون المصرف القاعدي . العنوان النسبي الموجود في التعليمات يُدعى إزاحة (déplacement) .

المكنات IBM 360/370 تتمتع بـ 16 مصرفاً عاماً يمكن أن تُستعمل كمصارف قاعدية . يُحدد المصرف بالكامل بواسطة رقم المصرف المستعمل كمصرف قاعدي والعنوان النسبي . هكذا ، فإن حقل العنوان من تعليمات هذه المكنات سيحتوي على حيز من أربع بتات حيث يتم تخزين رقم مصرف القاعدة .
الحسنات :

- يكتب المبرمج برنامجاً بشكل مستقل عن الموقع الذي سيشغله في داخل الذاكرة .
- البرنامج ، أو مجموعة الحيزات والتعليمات ، هو قابل للتحويل والنقل . من الممكن نقله من حيز من الذاكرة إلى حيز آخر دون تعديل في العناوين المنقولة (المحولة) . يكفي تعديل مضمون المصرف القاعدي .
- العنوان الأساسية وبشكل عام العنوان النسبية تسمح بعنونة مناطق كبيرة من الذاكرة بدلاً من أن تحتوي التعليمات ، على حقل عنوان طويل جداً . نشير حول هذا الموضوع ، أنه لعنونة 2^n خلية من الذاكرة يلزمنا عدد n من البتات .
السيئات :

- كل تعديل في مصرف القاعدة خلال تنفيذ التعليمات يؤدي إلى نتائج غير متوقعة .

3.3 . العنوان المؤشرة (Indexed address)

يتعلق ذلك بعملية حسابة العنوان بشكل شبيه بالعنونة القاعدية ولكن بهدف مختلف . يوجد مصرف يدعى مصرف التأشير أو مصرف الدليل (index register) ،

(1) العنوان القاعدي ليس بالضرورة عنوان زرع البرنامج .

تُخزَّن فيه قيمة معينة بواسطة المبرمج :
هكذا :

العنوان الفعلي = العنوان القاعدي + الإزاحة + مضمون المرصف المؤشر

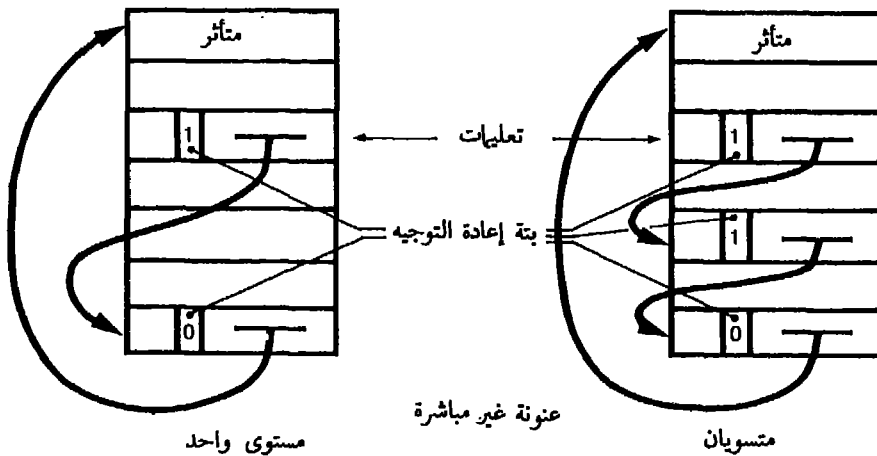
وعلى عكس مرصف القاعدة ، فإن مرصف التأشير يُمكن أن يُعدَّل مضمونه بواسطة المبرمج . هذه الأوالية تسمح ، بواسطة عمليات الزيادة على مضمونه هذاه بأن نقوم بعمليات تكرارية ، وتشكيل حلقات (loop) من التعليقات ، وبالتالي بلوغ خلايا متتالية من الذاكرة . هذه هي التقنية المستعملة لبلوغ الجداول . التعليقات التي تعود إلى عناوين والتي يُمكن أن تحتل عملية تأشير تتمتع بحقل إضافي خاص بالمرصف المؤشر حيث يستطيع المبرمج وضع رقم المرصف الذي يرغب باستعماله كدليل أو كمؤشر (index) .

4.3 . العنوان المباشرة

نتكلم عن العنوان المباشرة. عندما نجد في التعليمة العنوان الفعلي للمتأثر . إنها إذن أوالية العنوان البسيطة والمطلقة .

5.3 . العنوان غير المباشرة

هذه التقنية في العنوان موجودة على أكثر المكثات. حقل العنوان من التعليمة لا يحتوي على عنوان المتأثر ولكن على كلمة تحتوي عنوان المتأثر . بعض المكثات تتمتع ، عتادياً ، بأداة خاصّة لتغيير الاتجاه . في هذه الحالة ، يوجد بته خاصة في التعليمة تشير إلى وجود أو عدم وجود إعادة تغيير في الاتجاه . إعادة التوجيه يمكن أن تتم في مستويات عديدة كما يبرهن لنا المثل التالي :



6.3 . العنوان التلقائية

هَذَا المصطلح الشائع هو سيء لأن هذه الطريقة لا تخص عنواناً معيناً إنما تخص قيمة محدّدة . المعلومة الموجودة في حقل التعليمة المُستعمل لكتابة العنوان ، لا تُمثّل عنوان المتأثر ، إنما المتأثر نفسه (قيمة تستعملها التعليمة) .

تصغير المرصف يمكن أن يتم بطريقتين :

- بواسطة العنوان المباشرة يتم تصغير كلمة من الذاكرة بعنوان A ، وسنستعمل تعليمة لشحن المرصف بعنوان مباشر مع مضمون $R:A \rightarrow (A)$ ؛

- بواسطة العنوان التلقائية ، سيجري نقل القيمة صفر الموجودة في التعليمة على موقع العنوان إلى المرصف مع احتمال إزاحة البتة ذات الوزن الأكبر إلى اليسار إذا كان حجم حقل العنوان أصغر من حجم المرصف . العملية تتم بدون مساعدة أية خلية إضافية من الذاكرة . الحاسبات IBM 360/370 تتمتع بمجموعة من التعليمات ، تلك ذات الصيغة SI ، وتعمل بعنوان تلقائية .

4 هيكليّة الحاسبات 370 / IBM 360

لن نقوم - هنا سوى بإيجاز المميزات الضرورية الواجب معرفتها للبرمجة . بعض النقاط يمكن أن تعتبر حاجزاً أمام القارئ المبتدئ ، وستوضح له لاحقاً إلا أننا وجدنا من المفيد تحديدها منذ الآن .

1.4 . الذاكرة

الذاكرة هي معنونة بالبايتات (فقرة 1.5.2) . وسعتها القصوى هي 16777216 بايتة (2²⁴) . تُرقّم البايتات على التوالي بدءاً من الصفر تجري التعليمات على سلاسل من البايتات ، نصف كلمات (عناوين مزدوجة) من بايتين ، وعلى كلمات (عناوين تقبل القسمة على 4) من أربع بايتات وكلمات مزدوجة (عناوين مضاعفة لـ 8) من ثمان بايتات . تُرقّم بتات الكلمات من اليسار إلى اليمين من 0 إلى 31 .

2.4 . المراسف

تستعمل مراسف التحكم بواسطة نظام التشغيل لإدارة الذاكرة . وهي مبلوغة بواسطة تعليمات مميّزة وخاصة ، لن نتكلّم عنها .

المراسف العامة وعددها 16 ومُرَقّمة من 0 إلى 15 ، ويمكن أن تُستعمل :
- كمراسف قاعدية (أساسية) (ما عدا المراسف 0) ، وتحتوي على عنوان مطلق من 24 بتة من اليمين .

- مراسف دليلية (مرصّف مؤشّر) (index register) (ما عدا المراسف رقم 0) .
- مرصّف شحن (مركم) أو توسيع لمرصّف الشحن يستعمل لإجراء العمليات على التمثيلات الداخلية للأعداد بفاصلة ثابتة أو عمليات منطقية . بعض العمليات تحتاج إلى وجود مرصفين « متلاحقين » (الضرب مثلاً) . نستعمل عندئذٍ مراسف عامة متتالية ، الأول يكون إلزامياً برقم مزدوج . سنُسمي لاحقاً زوجاً من المراسف كهذا ، مرصفاً مزدوجاً . التعليمات التي تستعمل مرصفاً مزدوجاً لا تشير سوى إلى المراسف برقم مزدوج .

المراسف الأربعة المتحركة هي متخصصة في الحسابات الجارية على الأعداد الممثلة بفاصلة متحركة . وتحمل الأرقام 0 ، 2 ، 4 ، 6 .

هذه المراسف هي بطول 64 بتة ويمكن أن تحتوي على عدد طويل بفاصلة متحركة أو عدد بطول قصير من نفس النوع . يشغل العدد القصير بفاصلة متحركة البتات ذات الأوزان العالية ، وتهمل البتات الأخرى . والمراسف المستعملة لتخزين الأعداد الممثلة بفاصلة متحركة أو المراسف المتحركة يمكن أن تزاوج (2-0 و 6-4) بالنسبة للعمليات بالنسق الواسع (extended format) .

3.4 . الكلمة PSW (Program status word)

الكلمة PSW هي عبارة عن كلمة مزدوجة متعددة الأدوار . نجد فيها ، عند الانقطاع ، عنوان التعليمات التالية المطلوب تنفيذها . وتحتوي على نتائج عمليات المقارنة (كود - الشرط) ، ومعلومات عن بعض الحوادث (كود الانقطاع) . وتسمح بتنقيح حوادث الزيادة عن السعة (overflow) ، وتشير الى طريقة تشغيل الحاسب (الصيغة الرئيسية أو المُمَيَّزة أو صيغة المسألة) .

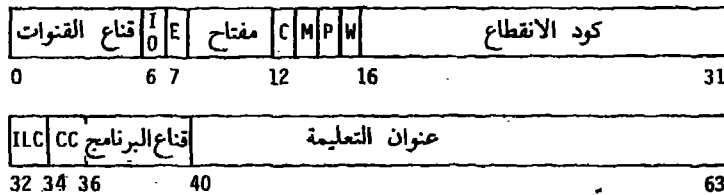
معرفة الكلمة PSW المرتبطة بالبرنامج تترجم إذا مفهومها الخاص بالتنفيذ . عند حدوث إنقطاع في البرنامج ، أي تعليق تنفيذه لمعالجة مسألة أكثر أولوية ، يتم تخزين الكلمة PSW الخاصة بالبرنامج المعلق في الذاكرة ، وتدعى عند ذلك الكلمة « PSW القديمة » . الكلمة PSW الجديدة ، والمرتبطة بالبرنامج الجديد الذي يعالج الانقطاع ، يتم شحنها مما يؤدي إلى تنفيذ برنامج جديد . البرنامج المعلق يمكن أن يُعاود تنفيذه بشرط ترميم أي إستعادة الكلمة PSW .

هناك طريقتان للتحكم موجودتان على المكنة 370 : الطريقة الأولى (Basic control mode) BC والطريقة EC (Extended control mode) .

وتختلف الطريقتان من حيث كون الترجمة الديناميكية للعنوان هي غير ممكنة سوى في الطريقة EC . وبكل طريقة في التحكم يرتبط نسق جديد للكلمة PSW . ونُمَيِّزها بواسطة البتة رقم 12 .

1.3.4 - الكلمة PSW في الطريقة BC (bit 12= 0)

هذا هو نسق الكلمة PSW على المكنات IBM 360 .



مخطط 1.4 . نسق الكلمة PSW في الصيغة BC

- الأقفنة . وهي مرتبطة بمختلف أسباب الانقطاعات . وجود البتة «0» في بتة القناع يمنع المعالجة المباشرة للحادثة . الانقطاعات من نوع overflow (قناع البرنامج) يمكن أن تهمل ، وتوضع الأخرى في الانتظار حتى رفع أو زوال سبب المنع أو الإهمال . فقط بإمكان المبرمج بلوغ قناع البرنامج عندما يعمل الأخير في صيغة المسألة (15=bit) البتة رقم 15 تعادل 1 .

البتات من 0 إلى 6 تتعلّق بالإنقطاعات الآتية من القنوات . البتة 7 (E) ، الانقطاعات الخارجية ، البتة 13 (M) ، عمل المكثف السيء والبتات من 36 إلى 39 ، الانقطاعات الناتجة عن تجاوز في السعة ، البتة 36 مرتبطة بالفيض عن السعة (Overflow) أثناء إجراء العمليات الجبرية بفاصلة ثابتة ، والبتة 37 متعلّقة بالنظام العشري والبتان 38 و39 متعلّقتان بالحساب بفاصلة متحركة .

- مفتاح الحماية : هذا المؤشر (البتات من 8 إلى 11) ، وبالعلاقة مع المفتاح الموجود في الذاكرة ، يتيح أو يمنع بلوغ البرنامج إلى بعض المناطق من الذاكرة . البتة 12 (C) تشير إلى طريقة العمل في التحكم $C=0$ تدل على طريقة العمل BC . البتة 14 (W) ، تساوي 1 عندما تكون الوحدة المركزية غير فعّالة ، في حالة الإنتظار (Wait) .

- البتة 15 (P) تعادل 1 عندما تكون الوحدة المركزية في الصيغة مسألة ، والتعليقات المميّزة هي أيضاً ممنوعة . وتعادل هذه البتة صفراً في صيغة العمل (Supervisor) أي المشرف .

- كود الإنقطاع : عندما يحدث أي إنقطاع ، فإن الكلمة القديمة PSW للبرنامج المقطوع تُخزّن في الذاكرة وتُوجد فيها كود خاص يُعرّف عن طبيعة الإنقطاع .

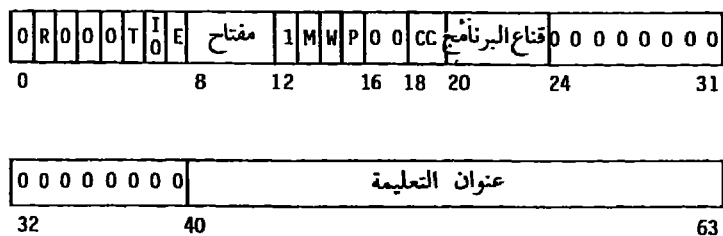
- ILC (البتان 32 و33) (Instruction Length code) . عند حدوث إنقطاع نجد في هاتين البتتين طول آخر تعليمة جرى تفسيرها .

- CC (البتان 34 و35) . عبارة عن الكود - الشرط الذي يعطي نتيجة المقارنة ، إشارة التأثير بعد تعليقات عديدة . . .

- عنوان التعليمة (البتات من 40 إلى 63) . هو عبارة عن عنوان التعليمة التالية المطلوب تنفيذها . تعرف PSW في لحظة الإنقطاع ، هذا الحقل يدل إذن على عنوان التعليمة حيث يجب أن يُعاود البرنامج عمله .

2.3.4 . الكلمة PSW في صيغة العمل EC (البتة 1=1) .
تختلف عن السابقة بواسطة إلغاء أقنعة القنوات ، وكود الإنقطاع والكود ILC .
ويستبدل ذلك بواسطة قناع «R» يدعى «program event recording mask» وبتة T

تتعلق بطريقة نقل العناوين . دراسة هذه الإمكانيات تخرج عن إطار هذا الكتاب ، ولن نتكلم عنها .



شكل 4.2 . النسخ PSW في الصيغة EC

5 لغة الآلة

1.5 . نسق التعليقات الآلية

لقد أدت بنا دراسة المكنة البسيطة إلى تعريف التعليقات الآلية بطول ثابت ،
والمركبة من كود للعملية ومن حقل للعنوان . تهتم العملية بتأثير واحد ، بينما يكون
التأثير الثاني موجوداً في مصرف الشحن أو المركم (Accumulator) .
تتمتع المكنات IBM 360/370 بأولية للعنونة أكثر تعقيداً ، تستعمل عدة مراصف
وتتمتع بـ 16 مصرفاً عاماً يُمكن أن تُستعمل كمراصف شحن . نرى إذاً أن تعليمة
بعنوان واحد ستكون مركبة من :

- كود للعملية (op. code) .
- رقم مصرف الشحن المعتمد في التعليمة .
- القسم عنوان الذي يتألف من :
- رقم المصرف القاعدي ،
- رقم المصرف الدليلي (المؤشر) إذا كان مستعملاً ،
- قيمة الإزاحة .

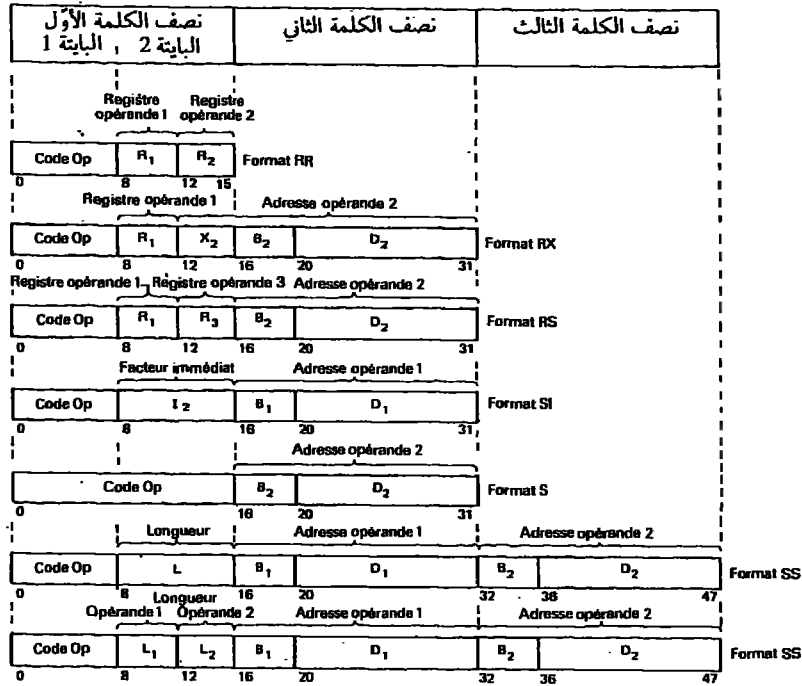
سيتم شرح تعليقات المكنات IBM 360/370 بواسطة ستة أشكال (نسق) مختلفة
تتعلق بطبيعة التأثيرات . التعليقات ذات النسق RR (Register to Register) لا
تستعمل سوى مصرفين . التعليقات من نوع RX تعالج عدداً موجوداً في أحد المراصف
وآخر على عنوان معين في الذاكرة وهذا العنوان يُمكن أن يكون دليلاً أو مؤشراً .
النسق RS (Register and Storage) ، و SI (Storage Immédiat) ،
و SS (Storage and Storage) ، و S لا تسمح بأي عملية تأشير.

الجدول التالي يُحدّد نسق التعليقات المستعمل . الحقول R ، X ، B ، D تُمثّل
على التوالي أرقام المراصف ، المراصف الدليلية ، مراصف القاعدة وقيمة الإزاحة .
الحرف L يرمز إلى طول التأثير ويُقاس بالبايتة في التعليقات بالنسق SS . الدليلان 1 و 2

يربطان هذه المعلومات بالمتأثر الأول والثاني .

سلاحظ إن البايته الأولى تحتوي دائماً على كود العملية (ما عدا بالنسبة للنسق S الذي يستعمل 2 بايته) ، إن نصفي الكلمة الثاني والثالث هما عبارة عن عناوين بشكل قاعدة وإزاحة . من المهم أن نتذكر أن التعليقات يجب أن تكون محصورة في نصف كلمات .

تتمتع التعليمة من نوع RX التي تستعمل عنواناً غير مؤشر بحيز X₂ يعادل الصفر . والتعليمة التي تستعمل عناوين غير مرتكزة على قاعدة سيكون فيها الحيز B صفراً . وبالتالي : فإن المرصف 0 لا يُستعمل لا كدليل ولا كمرصف قاعدي .



جدول 1.5

كود العملية المبتدئ بـ

الطول بالبايتات

Code opération commençant par	النسق Format	Longueur en octets
00	RR	2
01	RX	4
10	RX أو S ، SI ، RS	4
11	SS	6

جدول 2.5

وفي النهاية ، يُمكن أن نُذكر بأن البتين رقم 1 و2 من كود العملية ترمزان إلى طول ونسق التعليمة . الجدول 2.5 يوجز لنا ذلك .

2.5 . فئات التعليقات

من الممكن تصنيف التعليقات الآلية ضمن ست فئات :

1 - تعليقات التبادل :

- من مصرف إلى مصرف .
- من الذاكرة إلى مصرف (شحن المصرف LOAD) .
- من مصرف إلى الذاكرة (STORE) .
- من الذاكرة إلى الذاكرة .
- شحن تلقائي لأحد المراسف .
- شحن تلقائي للذاكرة .

2 - التعليقات الحسابية :

- الجارية على أعداد بالنظام الثنائي البحث (فاصلة ثابتة) ،
- على أعداد بفاصلة متحركة ، بدقة بسيطة ، بدقة مزدوجة أو بنسق موسّع ،
- على أعداد بالنظام العشري المكثف ،
- عمليات المقارنة الحسابية .

3 - التعليقات المنطقية :

- التقاطع ، الاتحاد ، المكاملة ...
- المقارنة المنطقية .

4 - تعليقات التحكم بتوالي التعليقات (تعديل مضمون عداد البرنامج PC) .

- تفريع إلزامي . .
- تفريع مشروط .

5 - تعليقات الإدخال / الإخراج (Input / Output)

6 - تعليقات متفرقة :

- تحويل النسق ، إختيار PSW ، الإزاحة ...

هذه التعليقات تعالج كلمات ، نصف كلمات ، كلمات مزدوجة أو سلاسل من السّماء . إضافة لذلك نجد عدة تعليقات للجمع حسب طول المتأثرات ، ومواقعها في الذاكرة أو في المراسف ، أو حسب تكويدها الداخلي . مجموع التعليقات يتجاوز إذاً 150 تعليمة .

3.5 . كتابة البرنامج بلغة الآلة

هدف هذا المثل هو الإعتياد على نسق التعليقات الآلية . نقترح جمع مضمون

- حيز الدليل (index) 0 (بدون تأشير)
- مصرف القاعدة F (المصرف 15)
- الإزاحة 0

أي :

5	8	2	0	F	0	0	0
COP	R ₁	X ₂	B ₂		D ₂		

تمثل المعطيات بفاصلة ثابتة ، سنستعمل التعليمه بكود العملية 5A التي تؤمن جمع مضمون الخلية ذات العنوان $B_1 + X_2 + D_2$ إلى المصرف المذكور في الحيز R_1 أي :

5	A	2	0	F	0	0	4
COP	R ₁	X ₂	B ₂		D ₂		

004 = إزاحة المتأثر الثاني بالنسبة إلى القاعدة .

وفي النهاية ، سنخزن النتيجة (التعليمه STORE ، بالكود 50) في الكلمة الثالثة على العنوان 8 .

5	0	2	0	F	0	0	8
COP	R ₁	X ₂	B ₂		D ₂		

بإمكاننا أن نفحص صورة البرنامج بعد تخزينه في الذاكرة .

العناوين الموجودة هنا هي العناوين النسبية ولا تتأثر بالعنوان الفعلي لموقع تخزين البرنامج . عنوان الاطلاق في التنفيذ ، أي عنوان أول تعليمه للتنفيذ ، هو عنوان القاعدة + C .

0	0	0	0	0	0	0	1	D
4	F	F	F	F	F	F	F	D
8	0	0	0	0	0	0	0	0
C	5	8	2	0	F	0	0	0
10	5	A	2	0	F	0	0	4
14	5	0	2	0	F	0	0	8
18								

عنوان الاطلاق

في التنفيذ

الحل الثاني :

إشحن المتأثرين الأول والثاني في المرافف ، وقم بعملية جمع لمضمون مرصف مع المرصف الآخر ومن ثم خزّن النتيجة . نستعمل المرافف 2 و 3 كمرافف للعمل والمرصف رقم 15 كمرصف قاعدي . والبرنامج هو التالي :

0	0	0	0	0	0	1	D	
4	F	F	F	F	F	F	D	
8	0	0	0	0	0	0	0	
C	5	8	2	0	F	0	0	0
10	5	8	3	0	F	0	0	4
14	1	A	2	3				
16	5	0	2	0	F	0	0	8

شحن المتأثر الأول في R₂
شحن المتأثر الثاني في R₃
جمع في R₂
خزن النتيجة

هذا الحل يحتاج إلى تعليمة إضافية . سنلاحظ وجود تعليمة من نوع RR بطول 2 بايتة .

الحل الثالث :

الحلّ الثالث كان سيقوم على إجراء الحساب مباشرة في الذاكرة دون استعمال المرافف . وسيحتاج إلى وجود تعليمة بثلاثة عناوين (المتأثر الأول ، المتأثر الثاني والنتيجة) . إلا أن هذا النوع من التعليمات هو غير موجود هنا .

خلاصة

نلاحظ ، في الأمثلة المذكورة ، أن حيز المؤشر (index zone) غير المستعمل هو مصفّر تماماً كما ذكرنا في الفقرة 1.5 .

إن البرمجة بلغة الآلة تبدو معقّدة ودقيقة رغم بساطة المثل وعدم إتمامه . لهذا السبب لا نستعمل هذا النوع من البرمجة ونفضّل عليه مرونة لغة المؤول (الأسمبلر) .

6 . لغة المؤول ASSEMBLER

المثل البسيط الذي جري عرضه في الفصل السابق أثبت لنا جميع صعوبات البرمجة بلغة الآلة مع أنه جرى تبسيط كبير لعملنا باستعمال النظام السادس عشري بدلاً من النظام الثنائي .

في لغة الآلة ، فإن أكواد العمليات والعناوين هي رقمية . وكل تعديل في موقع المعطيات يؤدي إلى تعديل العناوين في التعليقات المتعلقة بها .
هذه الصعوبات أدت بالمصممين إلى تعريف لغة ، تُدعى المؤول (assembler) ، قريبة من لغة الآلة ولكنها سهلة الإستعمال مما يجعل ترتيبها في مصاف اللغات المتطورة .

1.6 . مميزات لغات التأويل

- 1- تتميز التعليقات بلغة المؤول بكود عمليات تذكيري . مثلاً : تعليمة شحن المرصف من خلال مرصف آخر تتمتع بكود رمزي هو LR (LOAD TYPE RR) ، وتمتاز تعليقات الجمع بكود رمزي يبدأ بالحرف A ...
- 2- بإمكان المبرمج أن يقوم بتحديد عناوين بواسطة أسماء رمزية ويقوم برنامج ترجمة المؤول إلى لغة الآلة بربط القيمة الرقمية المناسبة بهذه الأسماء .
- 3- تتمتع لغة المؤول ليس فقط بمجموعة التعليقات الآلية التي تتضمنها لغة الآلة ، ولكن ببعض التعليقات الخاصة الآلية التي تدعى (التوجيهات) (أو أشباه التعليقات Pseudo-Instruction) وبعض الماكرو تعليمات (macro-instructions) .

2.6 . تعريفات

تدعى تعليمة - آلية كل تعليمة مكتوبة بلغة المؤول ومترجمة إلى تعليمة واحدة فقط بلغة الآلة . يتناسب كود رقمي مع كود - العملية التذكيري . مثلاً ، عملية نسخ المرصف 12 في المرصف 3 ، تكتب بلغة المؤول على الشكل التالي :

LR 3, 12 (LR = Load type RR)

وتُترجم إلى لغة الآلة بواسطة :

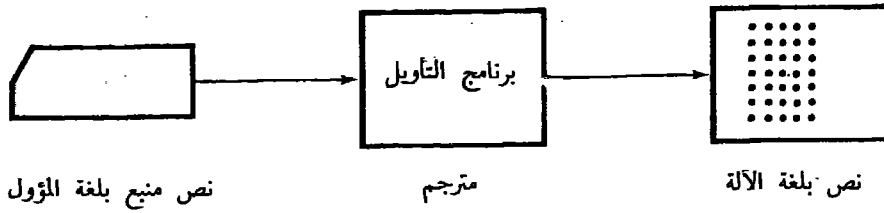
1	8	3	C
COP	R ₁	R ₂	

يُدعى أمر من نوع توجيه directive كل طلب إلى المؤول ، لا يُولّد أبداً تعليمة آلية ولكن يُقدم توجيهات للتأويل والتجميع . يوجد نوعان من التوجيهات : تلك التي لا تؤدي إلى أية عملية حجز للذاكرة وتلك المستعملة لحفظ موقع من الذاكرة أو تعريف الثوابت المفيدة للمسألة . هكذا ، فالتعليمة USING*,15 تعني إن المرفص 15 سيُعتبر أولاً كمرفص قاعدي ، مما سيسمح بعدم ذكر القاعدة (Base) في التعليمات التالية . هذا التوجيه لا يشغل مكاناً من الذاكرة في الكود المؤلّد ، وليس هو سوى إشارة إلى برنامج التأويل والتجميع . أن نكتب 'X'FOFO DC يعني أن نطلب إلى المؤول حجز بايتين من أجل تخزين الثابتة المحدّدة بالنظام السادس عشري بواسطة FOFO . لا يوجد توليد لتعليمة ولكن فقط حفظ لمكان من الذاكرة . من الممكن تشبيه هذه التوجيهات بتعليمات التصريح في اللغات المتطورة . أن نكتب بلغة فورتران الأمر dimension TAB (100) يعني أن نطلب من المصرّف (Compiler) حفظ المكان من الذاكرة اللازم لاستيعاب الجدول TAB (100) .

سنسمّي ماكرو - تعليمة (MACRO-INSTRUCTION) كل طلب إلى البرنامج المؤول assembler باستبدال سلسلة معرّفة مسبقاً من التعليمات تدعى ماكرو - تعريف . الماكرو تعريف هو إذاً عبارة عن مجموعة من التعليمات ينسخها البرنامج assembler مكان كل ماكرو - تعليمة . يقدم النظام مجموعة من الماكرو - تعريفات تدعى نموذجية (ستاندرد) تُسهّل على المبرمج القيام ببعض العمليات المعقدة ، كعمليات الإدخال - الإخراج . كما باستطاعة المبرمج أن يقوم بتعريف نظام خاص به من الماكرو - تعريفات .

3.6 عملية التأويل

الإسم «assembler» يعني في نفس الوقت اللغة والبرنامج الذي يقوم بترجمة النص إلى لغة - الآلة . سنقوم هنا بتناول مرحلة الترجمة بصورة موجزة . يبدو المؤول وكأنه عبارة عن مصرّف أو كأنه عبارة عن برنامج لترجمة النص المكتوب بلغة منبع إلى نص مستهدف يتألف من تعليمات - آلية . تدعى عملية الترجمة تأويلاً «assembling» .



1.3.6 . عداد المواقع

يجب على المؤول ، ومن خلال نص منبع ، أن ينتج نصاً ثنائياً يكون مع بعض التحويلات عبارة عن صورة البرنامج المطلوب تنفيذه . لتخصيص عناوين متتالية للتعليقات ، يستعمل المؤول عدداً للمواقع نرمز إليه بواسطة CE . في بداية عملية التأويل فإن CE يسبىء ، مثلاً يُصَفَّر . وخلال ترجمة التعليقات فإنه يزيد من قيمته حسب طول التعليقات المترجمة . وعندما يلتقي توجيهاً من نوع حجز لموقع أو منطقة من الذاكرة ، فإن مضمون CE يزداد حسب طول المنطقة المحجوزة . كل توجيه من نوع إشارة إلى برنامج التأويل لا يؤدي إلى زيادة في مضمون CE لعدم توليد أية تعليمة آلية . التعليقات ذات النسق RR تؤدي إلى زيادة مضمون CE 2 بايتة ، أما تلك التي تتمتع بنسق RX ، RS و SI فتؤدي إلى زيادة أربع بايتات إلى مضمون CE ، أما تلك ذات النسق SS فتؤدي إلى زيادة 6 إلى مضمونه . وكل توجيه لحجز كلمتين من الذاكرة يؤدي إلى زيادة مضمونه 8 بايتات .

في المثل التالي ، STARTO هي عبارة عن توجيه يؤدي إلى تهيئة CE وتصفيره . لا يحدث أي توليد لتعليقات جديدة وبالتالي فإن CE يبقى صفراً . STM 14, 12, 12(13) هي عبارة عن تعليمة من نوع RS تؤدي إلى زيادة 4 إلى مضمون CE . والتوجيه DS 1F يؤدي إلى حفظ كلمة من الذاكرة يُرمز إليها بواسطة ALPHA . وCE تزداد قيمته 4 بايتات . التعليمة LR 0,1 بالنسق RR تجعل مضمون CE يزداد 2 .

ملاحظات	منطقة التأثيرات كود العملية	العنوان الرمزي	CE بالنظام السادس عشري
تصفير CE تعليمة من نوع RS	START 0 STM 14,12,12(13)		0 0 4
حجز كلمة	DS 1F	ALPHA	20 24
تعليمة من نوع RR	LR 0,1	DEBUT	48 4A

وبالاختصار ، فإن عداد المواقع هو عبارة عن كلمة - ذاكرة يُخزّن فيها المؤول :
 قبل تأويل التعليم ، عنوان بداية التعليم (المتعلّق بتهيئة CE) ،
 - بعد التأويل ، عنوان الخلية الأولى المتوفرة .
 من الممكن أن نلاحظ إن قيمة CE تعادل قيمة مضمون عداد البرنامج عند التنفيذ .

2.3.6 . العنونة الرمزية والمرجعيات المطلقة

لقد ذكرنا سابقاً أنّ أحد أهم مميزات وخصائص المؤول تكمن في إمكان تسمية العناوين والقيم بواسطة رموز . يمكن أن يكون الرمز عبارة عن إسم منطقة من الذاكرة . في الجدول السابق ، فإنّ ALPHA و DEBUT هما عبارة عن عنوانين رمزيين نستطيع بلوغهما والعودة إليهما . سيكون بإمكان المبرمج أن يراجع مناطق من الذاكرة تبعاً لهذين العنوانين بواسطة تعابير من نوع $DEBUT - 2, ALPHA + 8$.

يُستعمل الرمز * لتسمية القيمة التي يأخذها CE في لحظة التأويل ، أي عنوان البايته اليسرى من التعليم الموجودة في طور التأويل . من الممكن أن نعود أيضاً بواسطة 2- * إلى العنوان الجاري ناقص 2 بايطة .

سنلاحظ أيضاً أنّه لا يمكن لقيمتين مختلفتين لمضمون CE أن تحملا نفس الإسم . إذ نكون عندئذ في حالة التعريف المزدوج .

يسمح المؤول أيضاً ببلوغ قيم مطلقة بواسطة رموز ، أي رموز غير متغيرة عند ترجمة البرنامج . تكتب عملية نسخ المرصف 1 في المرصف 0 مثلاً : $LR\ 0,1$. يمكننا أيضاً أن نكتب ، بشكل أوضح $LR\ R0, R1$ بشرط تحديد كون $R0$ و $R1$ عبارة عن رمزين مطلقين يعادلان القيمتين 0 و 1 .

وفي النتيجة ، فإن المؤول سيربط بكل رمز قيمة تدعى قيمة - خاصة ، وهذه القيمة سيتم ترجمتها أو عدمه حسب الحالة .

3.3.6 . جدول الرموز

عند العمل ، وفي كل مرة يلتقي المؤول رمزاً معيناً في منطقة الوسم (Label) يقوم بتخصيص خاصيات له :

- خاصة - قيمة تعادل قيمة CE في هذا الموقع .
 - خاصة - طول تعادل البعد (الحجم) بالبايتات للمنطقة المعينة .
- يمكن أن يقوم المؤول إذاً ببناء جدول من الرموز على الشكل التالي :

خاصية - طول	خاصية - قيمة	وسم رمزي
4	20	ALPHA
...	...	BETA
2	48	DÉBUT
...

عندما يلتقي رمزاً معيناً في قسم العنوان من التعليم ، يقوم المؤول باستشارة هذا الجدول . فإذا كان هذا الرمز موجوداً فيه معنى ذلك أن الرمز محدد مسبقاً ، وإلا فذلك يعني مرجعاً إلى الأمام ، أي إنه لم يلتقِ الرمز حتى الآن في منطقة الوسم ولكنه سيكون لاحقاً (إلا إذا كان يتعلق ذلك برمز خارجي ، أنظر الفصلين 20 و 21) .

4.3.6 . تأويل التعليم

يتعلق ذلك باختيار كيفية ترجمة التعليم بواسطة المؤول وبالأخص كيف يقوم بتحويل العنوان الرمزي الى عنوان قاعدي ، مؤشر وإزاحة . سنقوم بتحليل ذلك من خلال مثل معين .

لنفترض التعليم التالية :

L 12, ALPHA
العامل العامل
الثاني الأول

إنها تعليم من نوع RX ويكود عملية 58 (أنظر الملحق) حيث معناها هو « شحن مضمون الخلية ذات العنوان ALPHA في المرصف رقم 12 » . يقوم عمل المؤول على تعبئة مختلف حقول التعليم بالنسق RX ، أو :

5	8	C	0		
COP	R ₁	X ₂	B ₂	D ₂	
					منطقة العنوان

فلنلاحظ منذ الآن إن منطقة الدليل هي صفر ، لأنه لم يذكر أي مرصف مؤشر أو دليل في العامل الثاني من التعليم (الحقل الثاني منها) . ولتكملة حيز العنوان - يجب :

- معرفة المرصف المستعمل كقاعدة ،
- معرفة إزاحة العنوان ALPHA بالنسبة للعنوان القاعدي .

ونشير إلى أن العنوان القاعدي لا يختلط بالضرورة مع عنوان وجود البرنامج في الذاكرة .

سنقوم باقتراض في المثل إن ALPHA تناسب القيمة 1C للعداد CE ، وإن المرصف 15 هو مرصف القاعدة وإن العنوان القاعدي يناسب القيمة C للعداد CE . إزاحة ALPHA بالنسبة إلى القاعدة هي إذا C-C 1 أي 10 . التعليمة الآلية المؤولة ستكون إذا :

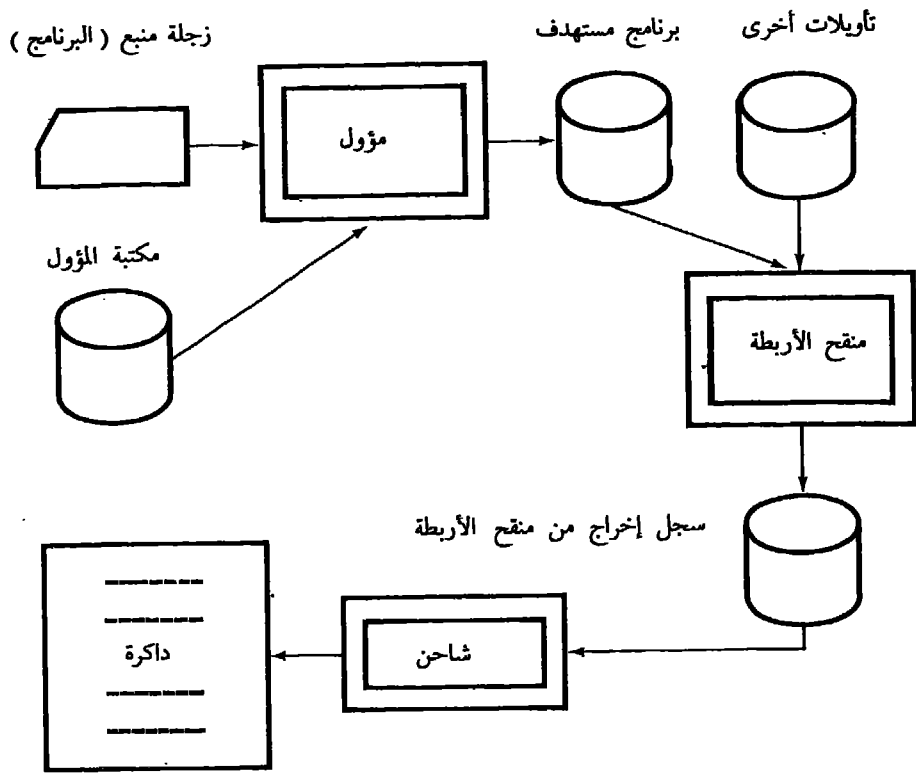
5	8	C	0	F	0	1	0
---	---	---	---	---	---	---	---

4.6 . مراحل تنفيذ البرنامج

إن تنفيذ البرنامج المكتوب بلغة المؤول ، كما بالنسبة للبرنامج المكتوب بإحدى اللغات المتطورة ، يتطلب عدة مراحل . المرحلة الأولى هي مرحلة التأويل والتجميع التي تكلمنا عنها . يُترجم النص الأولي إلى لغة الآلة ويُنسخ في سجل على الاسطوانة المغناطيسية . المرحلة الثانية ، التي يمكن أن تكون اختيارية للبرامج البسيطة ، هي تنقيح الأربطة (link editor) . وتؤدي إلى إجراء بعض الوصلات بين مختلف الزجل المؤولة بشكل منفصل أو التي تشكل جزءاً من مكتبة البرامج . منقح الأربطة يُشكل زجلة واحدة مستهدفة ، يمكن أن تتمتع بهيكلية تغطية ، من خلال مختلف عمليات التأويل . المرحلة التالية تقوم على شحن الزجلة في الذاكرة ، أي إعطائها عنواناً فعلياً لحزنها . وفي هذه الحالة تكون العناوين القاعدية متجمدة ، وبعض المعلومات المتعلقة بالعناوين المطلقة يجب أن تُحسب من جديد . يكفي إذاً أن نقوم بتخزين عنوان أول تعليمة للتنفيذ في عداد البرنامج CO (الكلمة الثانية من PSW) للبدء بمرحلة التنفيذ .

سنسمي نقطة الشحن أو عنوان الحزن ، عنوان بداية المنطقة المخصصة للبرنامج . سيُدعى عنوان الإطلاق عنوان أول تعليمة للتنفيذ من البرنامج . نقاط الدخول إلى البرنامج هي عناوين ، التعليمات أو المعطيات ، من الممكن بلوغها من خارج البرنامج . تتصل نقاط الدخول هذه بمنقح الأربطة الذي يمكن أن يقوم بإجراء وصلات بين مختلف الزجل (modules) . عنوان الإطلاق هو نقطة دخول .

بدون إعطاء جميع الإمكانات فإن المخطط 1.6 يعرض مختلف المراحل الواجب أن يتبعها البرنامج كي يجري تنفيذه .



خطط 1.6

القسم الثاني

المؤول 360 / 370

7 . العناصر الأساسية

1.7 . عموميات وتقديم البرنامج

- 1 - مجموعة السمات : يستعمل المؤول السمات الأبجدية A ، B ، Z... ، @ ، \$ ، والأرقام 0 ، 1 ، 2 ... 9 ، والسمات الخاصة : + - * / = () ، . ، ' ، & والقسمة البيضاء (blank) .

2 - ورقة البرنامج

المنطقة المحجوزة للمؤول تمتد من العامود 1 إلى العامود 71 . المنطقة 73 إلى 80 لا تُفسّر من جانب المؤول وتُستعمل لتعريف التعليقات . العامود 72 يُستعمل عندما ترغب إحدى التعليقات بالمتابعة على السطر التالي . تقسّم منطقة التعليمة (1 إلى 71) إلى أربعة أقسام :

منطقة الرموز : وتُستعمل لاجراء تخصيص رمزي للتعليمة (وسم) أو إلى معطى (اسم المعطى) .

الاسم المُخصّص :

- يبدأ بالعامود 1 بواسطة سمة أبجدية .
- يحتوي على أكثر من 8 سمات أبجدية .
- لا يحتوي على فراغ أو سمات خاصة .

الرموز التي تظهر في منطقة المتأثرات تخضع لنفس القواعد :

أمثلة :

2.7 . عناصر لغة المؤول

لقد لاحظنا حتى الآن إن المؤول يسمح لنا باستعمال رموز معينة لتسمية العناوين أو القيم . وعملياً فإن لغة المؤول تسمح لنا :

- باستعمال كتابات مثل '1011 B' ، 'A10C X' والتي ستعامل وكأنها قيم باللغة الثنائية ، أو السادس عشرية ... وهي ستكون عبارة عن القيم المعرفة أوتوماتيكياً .
- بلوغ الطول المتعلق بأحد الرموز . لو افترضنا إن «BIDON» هو وسم تعليمية ، أو بشكل عام ، أكثر اسم حيز معين ، فإن L'BIDON سيحدد طول التعليمية أو المنطقة . ويتعلق ذلك بالخاصية - طول ؛

- استعمال الأحرف كمناثرات في التعليقات ؛
- خلط كل هذه الإمكانيات لنحصل على تعابير ستكون معادلة لعناوين قابلة للنقل إلى قيم مطلقة .

من الملائم إذاً تحديد القواعد النحوية التي تسمح باستعمال هذه الإمكانيات

1.2.7 . قيم المعرفة الأوتوماتيكية (Auto-definition)

قيمة المعرفة الأوتوماتيكية هي واحد من أشكال الكتابة ، معروف من قبل المؤول ، يسمح بتحديد القيمة .

مثلاً :

'B' X ، '1011 B' و 11 هي عبارة عن ثلاث كتابات مختلفة تسمح بتحديد القيمة 11 (عشري) المثلة في المكنة بواسطة تشكيلة البتات 1011 . هذا الشكل في الكتابة هو مسموح ، مع بعض التحديدات ، بداخل حيز العوامل (منطقة العنوان) من التعليمية .

هناك أربعة أنواع من المعرفة الأوتوماتيكية المقبولة :

- الثنائي : '1001101 B' وعلى الأكثر 32 رقماً ثنائياً تحت إشراف النظام OS و 24 بالنظام (DOS) .

- السادس عشري : '1A3BC X' ،

- العشري : 125 (حد أقصى 10 أرقام عشرية) .

- نوع السمات : 'A' C ، 'C' (سمة أبوستروف أو الفاصلة العليا) ، 'ABCD C' ، 'CAB' . يجب أن نحصل كحد أقصى على أربع سمات بالنظام OS وثلاث بالنظام

DOS .

وبشكل عام ، فإن قيمة التعريف الأوتوماتيكي يجب أن تتم على 24 بته بإشراف النظام DOS وعلى 32 بته كحد أقصى بإشراف النظام OS . سنجد أمثلة على طرق استعمالها في الفقرة 3.7 المتعلقة بالتعابير .

2.2.7 . المتأثرات الحرفية

هي عبارة عن قيم مستعملة كمتأثرات في حيز عوامل التعليمات .
لشحن القيمة 125 في المرصف 3 يمكن للمبرمج أن يختار أحد حلين :

1- حجز حيز من الذاكرة ، يدعى ALPHA مثلاً ، ويُعرف عنه وكأنه يحتوي على القيمة 125 ، وبعد ذلك يُشحن ALPHA في المرصف 3 بواسطة التعليمات :
L3, ALPHA;

2- كتابة التعليمات : $L3, = F'125'$ ، وسيهتم المؤول بمحجز الخلية من الذاكرة التي تحتوي على 125 في منطقة تدعى POOL (حوض) .

في المثل المذكور لاحقاً ، فإن القارئ سيتحقق :

- من أن المؤول سيضع عنوان المتأثر الحرفي بشكل قاعدة وإزاحة داخل كود التعليمات المؤلد عنه ،

- من أن إستعمالين مختلفين لنفس المتأثر الحرفي لن يؤديا سوى إلى حجز واحد في الذاكرة ،

- من أن المتأثر الحرفي هو شبيه برمز قابل للترجمة .

إن استعمال المتأثر الحرفي ، إن لم يحمل أي شيء جديد ، فإنه يُقدم لنا فائدة بالنسبة لوضوح كتابة التعليمات .

قواعد الكتابة

يُحدد المتأثر الحرفي وكأنه متأثر عادي في توجيه DC مسبوق بالإشارة « = » . أما القواعد المتعلقة بتأثرات التوجيه DC فإنها ستوضح لاحقاً .

- لا يمكن أن يُستعمل المتأثر الحرفي كمعامل في التعبير (فقرة 3.7) الرقمي أو غير الرقمي .

- من البدني ، لأن المتأثر الحرفي يُستعمل « كمعطى للإدخال » في التعليمات ، أن لا يظهر في الحقل المُستقبل من التعليمات . سيكون من المتنافر أن نكتب : $ST3, = F'125'$ ($ST =$ خزّن مضمون المرصف في الذاكرة) .

L'OC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000000				1	CSECT
				2	EXTRN SPI
			00000	3	USING *.15
000000	5810 F018	00019		4	L 1.=F'0'
000004	5820 F01C	0001C		5	L 2.=C'ABCD'
000008	5820 F018	00018		6	L 2.=F'0'
00000C	5820 F020	00020		7	L 2.=V(SPI)
000010	5830 F01C	0001C		8	L 3.=C'ABCD'
000014	5810 F024	00024		9	L 1.=A(SPI)
				10	END
000018	00000000			11	=F'0'
00001C	C1C2C3C4			12	=C'ABCD'
000020	00000000			13	=V(SPI)
000024	00000000			14	=A(SPI)

3.2.7 . الخاصية - طول

وتسمح ببلوغ الطول المرتبط بالرمز . ويكتب :

مثلاً : L ' symbolic name L ' اسم رمزي ' L

L'ZONE L'SUITE L'*

- إذا كان الرمز هو اسم الحيز ، فهو يأخذ كقيمة طول الحيز مُقاساً بالبايتة .
 - إذا كان الرمز هو اسم التعليمة ، فهو يأخذ واحدة من القيم 2 ، 4 أو 6 حسب نسق التعليمة .

- إذا كان الرمز هو « * » ، فهو يأخذ كقيمة طول التعليمة التي يظهر فيها .
 بالنسبة للتوجيهين DC وDS . فإن الخاصية - طول لا تتأثر بوجود عامل الإزدواجية . سنلاحظ أنه بالنسبة للتوجيه EQU فإن قيمة الخاصية - طول هي قيمة المتأثر الأيسر .

الأمثلة التالية ، وللفهم الكامل ، تتطلب بأن نكون أكثر تقدماً في هذه الدراسة .
 إلا أننا نعرضها هنا :

الرمز	كود - العملية	عوامل	خاصية	قيمة
ZONE1	DS	CL80	L'ZONE1	80
ZONE2	DS	CL200	L'ZONE2	200
CARAC	DC	C'ABCDE'	L'CARAC	5
ABSOL1	EQU	ZONE2-ZONE1	L'ABSOL1	200
ABSOL2	EQU	25	L'ABSOL2	1
INSTR1	LR	0,1	{ L'INSTR1 , L'*	2 2
INSTR2	MVC	ZONE2(L'*),ZONE1	{ L'INSTR2 , L'*	6 6
	MVC	ZONE2(L'ZONE2-10),ZONE1	L'ZONE2	200
ALPHA	DC	6F'0'	L'ALPHA	4

3.7 التعابير

تعريف :

التعبير هو تركيب من الرموز ، وقيم التعريف - الأوتوماتيكي وخصيات - الطول في منطقة التأثيرات من التعليم .

الاستعمال :

تستعمل التعابير لتحديد :

- العنوان ،
- الطول الواضح ،
- المعدل ،
- عامل التكرار .
- التأثير .

فئات التعابير

التعابير هي بسيطة أو مركبة ، مطلقة أو قابلة للترجمة - التغير البسيط هو الرمز الوحيد أو الرمز « * » (قيمة عدّاد المواقع عند تأويل التعليم ، فقرة 1.3.6) .
التعبير المركّب هو مجموعة من عدة تعابير بسيطة مرتبطة بمؤثرات من نوع + ، - ، * ، (1) أو / ، التي تُمثل على التوالي الجمع ، الطرح ، الضرب والقسمة .
أمثلة :

ALPHA+2	**3	*	=	CE
ALPHA-BETA	*-2	*	=	CE
3*DELTA	A*3	*	=	مؤثر
(ALPHA-BETA)/2	**2			تعبير غير صالح
ALPHA+X'1A'	*3			تعبير غير صالح
TAB+L'LIGNE				

قواعد الإنشاء

التعبير المركّب :

- لا يمكن أن يبدأ بمؤثر ،
- لا يمكن أن يحتوي على مؤثرين ثنائيين متتاليين ،

(1) يجب عدم الخلط بين المؤثر * والرمز الذي يمثل عدّاد المواقع .

- لا يمكن أن يحتوي على نجمتين ،
 - لا يمكن أن يحتوي على تعبيرين بسيطين يتتابعان بدون مؤثر بينهما ،
 - لا يمكن أن يحتوي على متأثر حرفي .
- النظام OS يسمح باستعمال 19 مؤثراً أحادياً وثنائياً و6 مستويات من الأهلة . بينما النظام DOS لا يسمح سوى بـ 15 مؤثراً و5 مستويات .
- تقييم التعابير

يقوم المؤول بتخصيص قيمة رقمية لكل تعبير بسيط وبعد ذلك يُقِيم من اليسار إلى اليمين التعبير حسب أولوية خاصة للضرب وللقسمة بالنسبة للجمع والطرح .
 $A+B * C$ تُقِيم وكأنها $A+(B*C)$ وليس كأنها $(A+B) * C$. النتيجة الحسابية تصبح قيمة التعبير ، والمؤول يُقِيم بشكل طبيعي في المكان الأول المؤثرات الأحادية وداخل الأهلة . القسمة على صفر هي صحيحة وتعطي نتيجة صفر .

تعابير مطلقة ، تعابير منقولة
 التعبير المنقول هو تعبير حيث القيمة تتغير مقدار n إذا كان البرنامج منقولاً إلى n بايتة

التعبير المطلق هو التعبير الذي لا تتغير قيمته عند النقل .
 أمثلة :
 لنفترض إن ALPHA و BETA هي رموز منقولة وإن VAL1 و VAL2 هي رموز مطلقة :

تعابير مطلقة	تعابير منقولة
VAL1+B*101'	ALPHA+3
ALPHA-BETA	BETA+L'ZONE
VAL1+VAL2	BETA+VAL1

- التعبير سيكون مطلقاً إذا كان يحتوي على :
- رموز مطلقة ، قيم تعريفات أوتوماتيكية ، خاصيات - طول ،
 - رموز منقولة يظهر كل اثنين منها على حدة وتؤدي إلى تصفير فاعلية النقل .
- سنلاحظ إنه إذا كان $T1$ و $T2$ تعبيرين منقولين ، فإن $T1+T2$ و $T1*3$ ليست لا مطلقة ولا منقولة .

ولنتأكد من ذلك يكفي أن نقوم بإجراء عملية نقل بـ 100 مثلاً :

$T1 + 100$	تصبح	$T1$
$T2 + 100$	تصبح	$T2$
$T1 + T2 + 100$	تصبح	$T1 + T2$
$T1 * 3 + 300$	تصبح	$T1 * 3$

التعابير لا تحتل نفس الإزاحة .

إستعمال التعابير هو بشكل خاص مفيد لأنه يسمح بتحديد العناصر حيث القيم هي قابلة للتغيير عند التأويل وذلك بشكل مُعاملات ومتغيرات (مثلاً صفحة 122 ، السطر 78 من البرنامج) . كل تعديل في قيمة المتغير من التعبير سيكون محسوباً من جديد بواسطة المؤول وليس بواسطة المبرمج ، مما يُسهّل عمل المبرمج .

8 توجيهات تعريف الرموز

لنأخذ هذه القطعة من برنامج بلغة فورتران :

```
DIMENSION TAB(100)
-----
DO 50 I=1,100
TAB(I)=I
-----
50
```

يطلب الأمر DIMENSION حجز 100 كلمة - ذاكرة مجموعة تحت إسم الجدول TAB . تدل القواعد الضمنية المتعلقة بنوع المَعْرِفَات أن هذا الجدول سيتألف من أعداد حقيقية ، أي مَكُوْدَة في التمثيل بفاصلة متحركة بدقة بسيطة . يعرف المَصْرُف بأنه يجب أن يستعمل ، لتوليد كود التعليمات الحسابية التي تبلغ TAB ، التعليمات الحسابية بدقة بسيطة .

وفي فورتران ، كما في جميع لغات البرمجة ، كل رجوع إلى مَعْرِف يفترض أن يكون الأخير معروفاً من المَصْرُف ، أي مُحَدَّدًا خلال البرنامج بواسطة نوعه (حقيقي ، صحيح ...) وطوله مُقَاسًا بالكلمات أو بالبايتات . وفي النهاية يُخَصَّص المَعْرِفُ TAB بخاصية - قيمة (قيمة المَعْرِف ستكون عنوانه) وبخاصية - طول (بعد الحيز المشار إليه بالبايتة) .

في لغة التآويل المسألة هي نفسها ، يجب أن يحدّد كل رمز بواسطة خواصّه . سنرى توجيهين DC وDS يسمحان بتعريف الثوابت وحجز مكان من الذاكرة ، والتوجيه EQU الذي يسمح بإجراء توازنات بين الرموز .

1.8 . تعريف الثابتة DC

كثير الإستعمال ، هذا التوجيه يسمح بحجز منطقة من الذاكرة تحتوي على القيمة المدعّوة ثابتة وبتسميتها بواسطة أحد الرموز . شكل هذا التوجيه هو التالي :

عامل	كود العملية	رمز
$dtm'c'$	DC	[وسم]

- الوسم هو الإسم الرمزي للثابتة وهو اختياري .
- d هو عامل الازدواجية ، وهو اختياري ، وإذا كان مهملًا فإن قيمته تعادل 1 . إنه يشير إلى العدد الذي يجب أن تولّد فيه الثابتة .
- t هو النوع ، يمكن أن يكون أحد الأكواد الموجودة في الجدول التالي :

الاصطفاف	الطول الضميني	نسق المكنة	نوع الثابتة	كود
بايتة		EBCDIC	سمة	C
بايتة		ثنائي بفاصلة ثابتة	سادس عشري	X
بايتة		ثنائي	ثنائي	B
كلمة	كلمة واحدة	كلمة ثنائية بفاصلة ثابتة	عشري	F
نصف كلمة	نصف كلمة	نصف كلمة بفاصلة ثابتة	عشري	H
كلمة	كلمة واحدة	فاصلة متحركة ودقة بسيطة	عشري	E
: كلمة مزدوجة	كلمتان	فاصلة متحركة ودقة مضاعفة	عشري	D
: كلمة مزدوجة	4 كلمات	فاصلة متحركة ودقة رباعية	عشري	L
بايتة		عشري موسع	عشري	Z
بايتة		عشري مكثف	عشري	P

جدول 1.8

- في المكنة تُحصر الثوابت في حدود البايطة ، نصف الكلمة ، الكلمة أو الكلمة المزدوجة حسب نوعها ما عدا في الحالة التي نُحدّد فيها طولها (أو نستعمل معدّلًا للطول) .
- m هو معدّل طول الثابتة ، ويمكن أن يكون :

- أ - معدّل طول ضمني يُكتب على شكل Ln حيث n هو عدد البايتات في التمثيل الداخلي . إنّ وجود معدّل للطول يُصَفّر قاعدة الاصطفاف الضمنية .
- ب - مُعدّل للحصر يُكتب على الشكل التالي : Sn .

معدّل الحصر يقوم بإجراء إزاحة لـ n بته إلى اليسار إذا كانت n إيجابية ، وإلى اليمين إذا كانت n سلبية . أي يقوم بإجراء ضرب أو قسمة صحيحة على 2^n . معدّل الحصر ، ويدعى أيضاً المقياس ، يُطبّق على الثوابت E ، D ، و L .

' c ' هي الثابتة المحددة بين فاصلتين عليين (') . الثوابت يُمكن أن تكون محدّدة بإشارة ، فاصلة عشرية وبأس (قوة) يُرمز إليه بالحرف E . الأمثلة التالية تُظهر لنا مختلف الإمكانيات . وهناك جدول في الملحق يُوجز لنا مميزات الثوابت .

LOC	SUBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000000				1	CONST CSECT
				2	PRINT DATA
				3	*****
000000	C1C2C3C4C5			4	*****
000005	000006			5	CONSTANTES DE CARACTERES PAS D'ALIGNEMENT PARTICULIER. LONGUEUR 256
000010	C1C2C3C4C5			6	DECALAGE A GAUCHE, TRONCATURE A DROITE
000015	000016			7	CHARACT
000020	000021			8	DC C'ABCDE,
000025	000026			9	DC C'ABCDE,
000030	000031			10	DC CL3, TRONCATURE A DROITE,
000035	000036			11	DC CL6, ARC, CADRAGE A GAUCHE
000040	000041			12	DC CL8, L' APOSTROPHE, UN SEUL APOSTROPHE ENGENDRE
000045	000046			13	DC CL3, A68B, MEME REMARQUE
000050	000051			14	DC 3CL4, ABCDE, REPETITION ET TRONCATURE
000055	000056			15	*****
000060	000061			16	*****
000065	000066			17	CONSTANTES HEXADECIMALES. CADRAGE A DROITE. TRONCATURE A GAUCHE.
000070	000071			18	HEXA DC X'F01A2,
000075	000076			19	DC XL5, 1A80C,
000080	000081			20	DC 3XL2, 1A1FBC,
000085	000086			21	*****
000090	000091			22	CONSTANTES BINAIRES. LONGUEUR MAXI 256 OCTETS ; CADRAGE A DROITE
000095	000096			23	COMPLETE PAR DES ZEROS A GAUCHE, ALIGNEMENT SUR L'OCTET.
000100	000101			24	BINAIRE DC B'10001101101C,
000105	000106			25	*****
000110	000111			26	TRONCATURE A GAUCHE,
000115	000116			27	DC B'1, 1000111001,
000120	000121			28	DC 3BL2, 110,
000125	000126			29	*****
000130	000131			30	CONSTANTES EN VIRGULE FIXE SUR UN MOT (F) OU UN DEMI-MOT (H),
000135	000136			31	ALIGNEMENT SUR LE MOT OU LE DEMI-MOT. LORSQU'UNE LONGUEUR EST
000140	000141			32	SPECIFIEE IL N'Y A PLUS D'ALIGNEMENT. LA CONSTANTE EST EN DECIMAL
000145	000146			33	HALFWORD DC H'5E3, 10E2, 17, 2,
000150	000151			34	DC H'53, 50,
000155	000156			35	DC H'400,
000160	000161			36	DC H'53, 50,
000165	000166			37	DC H'6,
000170	000171			38	*****
000175	000176			39	DECALAGE 3 BITS A GAUCHE (*8)
000180	000181			40	DECALAGE 3 BITS A DROITE (/8)
000185	000186			41	*****
000190	000191			42	ARRONDI SUPERIEUR
000195	000196			43	ARRONDI INFERIEUR
000200	000201			44	MODIFICATION LONG ET ALIGN.
000205	000206			45	DECALAGE A GAUCHE DE 2 BITS
000210	000211			46	*****

2.8 . ثوابت العنوان⁽¹⁾

إنَّ تعريف ثابتة - عنوان يعني حجز مكان من الذاكرة لتخزين عنوان أحد العناصر . نشير هنا إلى بعض المفاهيم الأساسية . العنوان الفعلي ، أي العنوان الحقيقي لأحد العناصر هو غير معروف إلا عند شحن البرنامج في الذاكرة . لذا فمن غير الممكن ، في مرحلة التأويل والتجميع ، أن يكون بتصرفنا العنوان الفعلي الخاص بالرمز . نبلغ الرمز بواسطة الإزاحة نسبة إلى مضمون مرصف القاعدة .

في بعض الأحيان يبدو من غير الممكن بلوغ أحد الرموز التي لا تنتمي إلى الزجلة التي تكون في طور المعالجة من قبل المؤول . هذه هي الحالة ، مثلاً ، عندما نرغب بإجراء تفريع إلى برنامج - ثانوي مؤول ومترجم على حدة . الحلّ يقوم إذاً ، بالنسبة للمؤول ، على بلوغ مباشر بسبب وجود كلمة ، تدعى ثابتة - عنوان ، يقوم الشاحن (Loader) بملئها بشكل مناسب .

مثلاً :

نرغب ، للتفريع إلى المرصف 15 ، شحنه بعنوان نقطة الدخول P1 لبرنامج - ثانوي مؤول على حدة . سنحفظ ، في الزجلة المتنادية ، كلمة تدعى هنا ADRP1 سيتم تعريفها كثابتة عنوان خارجية . والمؤول سيقوم بإعدادها وتصفيها ، كما سيقوم الشاحن بتخزين العنوان الفعلي P1 في داخلها . العنوان P1 سنحصل عليه إذاً في المرصف 15 بواسطة التعليمات :

. L 15, ADRP1

إنَّ نسق تعريف ثابتة العنوان هو التالي :

عامل	كود - العملية	رمز
d t m 1(c)	DC	[وسم]

نسق هذا الأمر لا يتميز عن نسق تعريف الثوابت إلا بتبديل الفواصل العليا بالأهلة .

- d هو عامل الإزدواجية ، وإذا جرى إهماله فإنّه يعادل 1 .
- t هو كود نوع الثابتة .

(1) دراسة هذه الفقرة المفيدة للفهم الكامل يمكن أن يقفز عنها عند القراءة الأولى .

وقد يكون A, Y, S, V أو Q (النوع Q ليس متوقفاً سوى تحت النظام OS). النوعان A و Y يسمحان بتعريف الثوابت بواسطة تعابير بسيطة أو مركبة ، مطلقة أو منقولة . القيمة ثابتة العنوان محدّدة لجهة اليمين في كلمة (نوع A) أو نصف كلمة (نوع Y) . الثوابت من نوع S تسمح بتخزين عناوين بشكل قاعدة وإزاحة على نصف كلمة . ولا يمكنها أن تعرّف في نص حرفي . تستعمل الثوابت من نوع V لتعريف عناوين خارجية من نوع « إسم برنامج ثانوي » .

- m هو عبارة عن معدّل الطول الضمني . وجود المعدّل يؤدي إلى إلغاء قاعدة الاصطفاف الأوتوماتيكية (alignment) .
- C هو عبارة عن الثابتة نفسها مكتوبة بدأخل أهلة . الأمثلة في الصفحة 75 تعرض وتعرّف كل نوع من الثوابت .

استعمال ثابتة العنوان :

تستعمل :

- لشحن عنوان في مصرف .
- لاجراء وصلات بين البرنامج والبرنامج الثانوي .
- وسيتم درسن ذلك في الفصلين 20 و 21 .

3.8 . أمر حجز مواقع من الذاكرة

هذا الأمر هو عبارة عن توجيه يسمح بحجز موقع من الذاكرة دون إعداد أو تهيئة مضمونه عند التأويل . هذا الأمر يؤدي إذاً إلى زيادة مضمون عداد المواقع . ويسمح بتسمية المناطق المحددة وبلوغها رمزياً . النحو ، القريب من نحو التوجيه DC ، هو التالي :

عامل	كود العملية	رمز
$d \ t \ m$	DS	[وسم]

- d مُعامل الازدواجية ، وهو اختياري . وإذا كان صفراً فهو يسمح بزيادة عداد المواقع حتى حدود نصف كلمة ، كلمة أو كلمة مزدوجة حسب نوع t المرتبطة بالمنطقة . هذه الخصوصية تستعمل كثيراً ونوضّحها في الأمثلة والأسئلة . سنشير هنا ، إلى أنه مع وجود عامل ازدواجية يعادل صفراً ، فإن الوسم الموجود في منطقة الرمز هو مخزن في جدول الرموز .

LDC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	SYMBOLE EXTERNE
000000				1	CSECT	
				2	EXTRN	
000000		00000		3	USING *	
		003E8		4	ORG *	
0003E8	FFFFFFFF	000FF		5	ORG *	
				6	ABSLC	
				7	RELOC	
				8	DC	
				9	CONSTANTES D'ADRESSE DE TYPE A	
				10	SECRET DC A(ABSOL+20)	
				11	SECRET DC A(ABSOL+20)	
				12	SECRET DC A(ABSOL+20)	
				13	SECRET DC A(ABSOL+20)	
				14	SECRET DC A(ABSOL+20)	
				15	SECRET DC A(ABSOL+20)	
0003EC	000003E8			16	SECRET DC A(ABSOL+20)	
0003FD	00000113			17	SECRET DC A(ABSOL+20)	
0003FA	000003FA			18	SECRET DC A(ABSOL+20)	
0003F8	7D16			19	SECRET DC A(ABSOL+20)	
0003FA	0000			20	SECRET DC A(ABSOL+20)	
0003FC	000003F2			21	SECRET DC A(ABSOL+20)	
000400	00000000			22	SECRET DC A(ABSOL+20)	
				23	CONSTANTES D'ADRESSE DE TYPE Y	
				24	SECRET DC Y(ABSOL+20)	
				25	SECRET DC Y(ABSOL+20)	
				26	SECRET DC Y(ABSOL+20)	
				27	SECRET DC Y(ABSOL+20)	
				28	SECRET DC Y(ABSOL+20)	
				29	SECRET DC Y(ABSOL+20)	
000404	00000002			30	SECRET DC Y(ABSOL+20)	
000408	7D			31	SECRET DC Y(ABSOL+20)	
000409	02			32	SECRET DC Y(ABSOL+20)	
				33	SECRET DC Y(ABSOL+20)	
				34	CONSTANTES D'ADRESSE DE TYPE S	
				35	SECRET DC S(ABSOL+20)	
				36	SECRET DC S(ABSOL+20)	
				37	SECRET DC S(ABSOL+20)	
				38	SECRET DC S(ABSOL+20)	
				39	SECRET DC S(ABSOL+20)	
				40	SECRET DC S(ABSOL+20)	
				41	SECRET DC S(ABSOL+20)	
00040A	0400			42	SECRET DC S(ABSOL+20)	
00040C	03E8			43	SECRET DC S(ABSOL+20)	
00040E	C200			44	SECRET DC S(ABSOL+20)	
				45	SECRET DC S(ABSOL+20)	
				46	CONSTANTES D'ADRESSE DE TYPE V	
				47	SECRET DC V(ABSOL+20)	
				48	SECRET DC V(ABSOL+20)	
				49	SECRET DC V(ABSOL+20)	
				50	SECRET DC V(ABSOL+20)	
				51	SECRET DC V(ABSOL+20)	
				52	SECRET DC V(ABSOL+20)	
				53	SECRET DC V(ABSOL+20)	
				54	SECRET DC V(ABSOL+20)	
				55	SECRET DC V(ABSOL+20)	
				56	SECRET DC V(ABSOL+20)	
				57	SECRET DC V(ABSOL+20)	
000410	00000000			58	SECRET DC V(ABSOL+20)	
000414	00000000			59	SECRET DC V(ABSOL+20)	
				60	SECRET DC V(ABSOL+20)	

- t يُحدّد نوع المنطقة أي بالتحديد كما جرى بالنسبة للأمر DC . وهو إلزامي ويحدّد التسطير الضمني .

- m هو معدّل الطول ويكتب Ln ، حيث n هو طول المنطقة بالبايتات . كما بالنسبة للأمر DC فهو إختياري . وجوده يلغي فعل الإصطفاف الضمني . سنشير هنا إلى أن الطول الأقصى للثابتة من نوع سلسلة السّيات المحدّدة في الأمر DC هو 256 بايتة ، وإستعمال النظام OS يسمح بـ 65535 بايتة .

لتسهيل صيانة البرامج سنستعمل : ETIQ DS0H لتعريف نقاط التفريع .
قدر المستطاع سنفضل إستعمال الأمر DC عن الأمر DS الذي يقوم بإعداد المنطقة بقيمة محايدة ستكون مرئية في عملية DUMP (دلق) .

4.8 . توجيه التعادل EQU

يسمح بتعريف رمز وإعطائه قيمة مطلقة أو محوّل ويكتب على الشكل التالي :

تعبير مطلق أو محوّل	EQU	رمز (Symbol)
---------------------	-----	--------------

سنشير هنا إلى أن وجود الرمز هو إلزامي . لا يحجز التوجيه أي موقع من الذاكرة ولا يقوم سوى بإنشاء رمز جديد في جدول الرموز . ويمكن أن يكون موجوداً في أي موقع من البرنامج ويستخدم :

1- لاستعمال أسماء بدلاً من القيم . تجري العادة مثلاً على كتابة :

R0	EQU	0
R1	EQU	1
...
R15	EQU	15

عما يسمح ، منذ البداية ، ببلوغ المرافف بواسطة الأسماء R0 ، R1 ، ... ، R15 بدلاً من القيم 0 ، 1 ، ... ، 15 . هذا ما يؤدي إلى فائدة ووضوح في العمل ولكن أيضاً إلى إمكانية إيجاد مراجع المرافف بسهولة لأنها ستظهر في جدول الرموز وفي البلوغ التصالي .

2- لتخصيص قيمة جديدة محدّدة داخل البرنامج لرمز معين ، أي معرّف خلال الأبطر السابقة .

RO	EQU	0	
REGO	EQU	RO	(رمز مطلق)
DEB	LR	R1,R2	
DEBUT	EQU	DEB	(رمز محوّل)
ZONE	DS	4F	
Z1	EQU	ZONE+12	(تعبير محوّل)

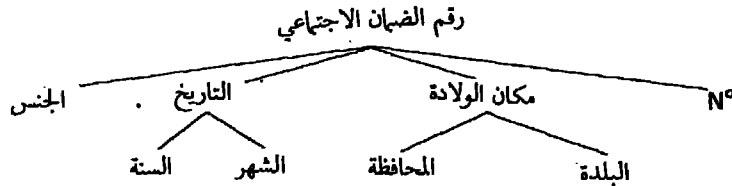
3- لحساب التعابير حيث القيمة مجهولة في لحظة الكتابة أو صعوبة الحساب وتخصيص رمز لها .

EXPRES EQU A-(B+C)/5-D
ETIQ EQU * (قيمة عدد المواقع)

تمارين

تمرين 1.8 - ولد ، بواسطة تعريف ثابتة مخصصة ، منطقة من الذاكرة بحجم 100 بايتة تحتوي على سلسلة من 100 عدد صحيح طبيعي . نفس السؤال لمنطقة بحجم 100 كلمة .

تمرين 2.8 - عرّف حجز من الذاكرة لاستيعاب رقم الضمان الاجتماعي (13 سمة) مع وصف للهيكلة التالية .



وذلك بفحص الخاصية - طول لكل معرف مذكور .

تمرين 3.8 - باستعمال الأمر ORG (فقرة 3.20) ، مطلوب تعريف منطقة من الذاكرة يمكن أن تستوعب إما ثمناً (8 أرقام عشرية موسّعة) أو كمية (4 أرقام عشرية موسّعة) ، أوروباً (عددًا صحيحاً بفاصلة ثابتة) ونصاً من 10 سمات . يتعلّق ذلك بإعادة تعريف من نوع REDEFINES بلغة كوبول .

9 كتابة العناوين بلغة المؤول

1.9 . قاعدة ضمنية ، قاعدة جلية

في جسم التعليقات الآلية ، فإن العناوين المحولة تكون ممثلة بواسطة مرصف قاعدي ، وإزاحة ومرصف دليل (حالة النسق RX) . عند كتابة التعليقات - الآلية بلغة المؤول سنقوم بإيجاد ثلاثة متأثرات . لقد لاحظنا حتى الآن أنه كان يوجد ستة أنسقة مختلفة للتعليقات الآلية . إضافة لذلك ، وفي لغة المؤول ، فإن كتابة منطقة العوامل (منطقة العناوين والثوابت) ستتغير حسب نسق المكنة .

لنأخذ تعليمة شحن المرصف 3 (LOAD) من خلال مضمون عنوان معين . لنفترض إن المرصف 15 قد جرى إختياره كمرصف قاعدي ، وإن العنوان موضع السؤال هو موجود على مسافة 512 (في القاعدة العاشرة) من العنوان القاعدي وهو مؤشر بواسطة المرصف 5 . التعليمة - الآلية سيكون لها الشكل التالي :

5	8	3	5	F	2	0	0
COP	R ₁	X ₂	B ₂		D ₂		

سيكون بإمكان المبرمج بلغة المؤول أن يكتب التعليمة على الشكل التالي :
(5, 15) 512, 3 L . القاعدة 15 هي هنا مسماة بشكل واضح . لا نرى بهذا الشكل الفائدة الرمزية من لغة المؤول .

لنأمن ببساطة أكبر فإن المؤول يسمح بعدم ذكر القاعدة في منطقة العوامل التابعة للتعليمة . يكفي لذلك أن نصرّح ، بواسطة التوجيه 15, * USING ، أن التعليقات التالية يجب أن تؤول (تجمّع) مع المرصف 15 كقاعدة . الفائدة الأولى هي السماح بتعديل مرصف القاعدة دون إعادة كتابة جميع التعليقات . كذلك ، فإن الإزاحة ومرصف المؤشر يمكن أن يتم تمثيلها بشكل رمزي عند الحاجة . هكذا ، فلنأخذ العنوان المحوّل ALPHA الموجود على المسافة 512 بايتة من العنوان القاعدي . ولنشحن في

المرصف 3 مضمون العنوان ALPHA المؤشر بواسطة المرصف 5 . بإمكاننا كتابة التعليقات التالية بلغة المؤول :

- بتحديد القاعدة بشكل واضح : $L 3,512 (5,15)$.
- أو $(5) ALPHA 3, L$ ، القاعدة هي ضمناً مرتبطة بـ ALPHA ومحددة بواسطه المؤول حسب التوجيه USING . يوجد عدة إمكانيات لكتابة منطقة العوامل ، وهذا ما سنقوم بشرحه الآن .

2.9 . كتابة العوامل

في الإعتبارات التالية D, X, R, B, M و L تمثّل على التوالي الإزاحة ، رقم مرصف المؤشر ، رقم مرصف القاعدة ، رقم المرصف العام ، قناع (موجود في التعليم) والطول . الدلائل 1 ، 2 و 3 هي مرتبطة بمختلف المتأثرات . جميع هذه الرموز يجب أن تكون عبارة عن تعابير مطلقة . S ستمثّل تعبيراً متحولاً يمكن أن يُختزل عملياً إلى رمز واحد . وبتحديد أكثر للمرصف القاعدي ، فإن عوامل (متأثرات) التعليقات يُمكن أن تُكتب بلغة المؤول ، حسب النسق ، على الشكل التالي :

النسق	المعاملات
RR	R_1, R_2
RX	$R_1, D_2(X_2, B_2)$
RS	$\begin{cases} R_1, R_3, D_2(B_2) \\ R_1, M_3, D_2(B_2) \end{cases}$
SI	$D_1(B_1), I_2$
SS	$\begin{cases} D_1(L, B_1), D_2(B_2) \\ D_1(L_1, B_1), D_2(L_2, B_2) \end{cases}$
S	$D_2(B_2)$

جدول 1.9

العنوان المحوّل هو دائماً العنوان المحسوب في لحظة تنفيذ الجمع $D+X+B$ للتعليقات ذات النسق RX أو $D+B$ للتعليقات RS ، SI أو SS . العوامل $D(X, B)$ ، $D(B)$ ، و $D(L, B)$ يُمكن أن تُستبدل بواسطة العوامل حيث رقم المرصف القاعدي والإزاحة سيتم حسابها بواسطة المؤول . وستُكتب إذاً على الشكل التالي : S ، $S(X)$ ، أو $S(L)$.

الجدول التالي يعرض لنا مختلف إمكانيات كتابة هذه المعاملات حسب نسق

التعليمة . منشير هنا إلى أنه بداخل الأهلة ، وفي الشككين مع قاعدة ضمنية أو جلية ، لا يمكن أن نجد سوى التعابير المطلقة حيث المعنى الأساسي ، الدليل أو الطول يتعلّق بنسق التعليمة وبالطبيعة مطلق أو محوّل للتعبير المذكور على يسار الأهلة .
أمثلة :

ABS و TRANS هما تعبيران مطلقان ومحوّلان . ABS1 (ABS2) في التعليمة RS يمكن أن تُفهم وكأنها D(B) . ABS1 (TRANS) هي مغلوبة مهما يكن النسق ، TRANS (ABS1) يجب أن تُفهم كأنها S(X) في التعليمة RX وكأنها S(L) في التعليمة SS .

نسق التعليمة	الكتابة بتعابير مطلقة قاعدة جلية	الكتابة بتعابير محوّلة قاعدة ضمنية
RS et SI	D(B)	S
SS	D(L,B) D(,B) (1) D(B)	S(L)
RX	D(X,B)	S(X) S

جدول 2.9

حالات خاصة

X أو B يعادل صفراً .

D(0) يمكن أن يُكتب D

D (0,B) يمكن أن يُكتب D(,B)

D (X,0) يمكن أن يُكتب D(X,) أو D(X) . (أمثلة أنظر صفحة 82) .

3.9 . قواعد الاصطفااف أو التراصف

مع أن أوالية العنونة تسمح بعنونة البايته ، فإن عناوين متأثرات التعليمة يجب أن تخضع لبعض قواعد التوافق . قواعد كهذه هي موجودة على جميع المكتات .

تستعمل التعليقات متأثرين قديكونان عبارة عن مرصف وعنوان من الذاكرة أو عناوين من الذاكرة . نحدّد القواعد حسب المعطيات التي تُعالجها التعليقات .

بالنسبة للتعليقات التي تُعالج كلمات - مزدوجة ، كلمات أو نصف - كلمات ، فإن

(1) الطول هو ضمني ، المؤل يختار الخاصية - طول . الطول المؤل هو دوماً الطول الفعلي ناقص واحد .

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000000				1	PRINT DATA
000000	90EC D00C	00000C		2	START 0
000004	05C0		000006	3	* SEQUENCE D'ENTREE
				4	STM 14,12,12(13)
				5	BALR 12,0
				6	USING *12
000006	47F0 C00E	000014		7	* ACBASE BC 15,INSTR1
			000000	8	RO 0
			000003	9	RO 3
			000005	10	RO 5
			00000C	11	R3 EQU 12
				12	R9 EQU 12
				13	R12 CNOP 0,4
				14	ALPHA DC X'01234567'
				15	DC X'89ABCDEF'
00000A	0700			16	* INSTR1 L 3,4(5,12)
00000C	01234567			17	* REGISTRES 3, L'ECRIURE (3) EST LA SEULE OUI SOIT INDEPENDANTE
000010	89ABCDEF			18	* DE LA PLACE DE L'ALPHA PAR RAPPORT A L'ACRESSE DE BASE.
000014	5835 C004	000004		19	R3,ALPHA+4
000018	5835 C004	000004		20	R3,ALPHA+4
00001C	5830 C00A	000010		21	* TOUTES LES INSTRUCTIONS (3) A (7) CHARGENT X'89ABCDEF' DANS LE
000020	5830 C008	000008		22	(3) EMPLOI D'UNE EXPRESSION
000024	5830 C008	000008		23	TRANSLATABLE. BASE IMPLICITE.
000028	5830 C008	000008		24	(4) INSTR. IDENTIQUE SYMB. ABSOL
00002C	5830 C052	000008		25	(5)
000030	5830 7008	000008		26	(6)
000034	5830 C007	00000D		27	(7) EMPLOI D'UN LITERAL
	*** ERROR ***			28	(8) "8" EST UN DEPLACEMENT
000038	5830 D00C	00000C		29	(9) ERREUR D'ALIGNEMENT
00003C	5830 C008	000000		30	
000040	0000 D000	000000		31	(10) "13" EST UN DEPLACEMENT
	*** ERROR ***			32	(11) ERREUR DE SYNTAXE
000044	0070 D000	000000		33	(12) ERREUR DE SYNTAXE
	*** ERROR ***			34	(13) ERREUR DE SYNTAXE
000048	000C 0000	000000		35	(14) ERREUR DE SYNTAXE
	*** ERROR ***			36	(15) 12 EST UN INDEX
00004C	583C 0000	000000		37	
				38	
				39	
				40	
				41	

STMT ERROR CODE MESSAGE ASSEMBLER DIAGNOSTICS AND STATISTICS

34 IF0220 ALIGNMENT ERROR NEAR OPERAND COLUMN 11

36 IF0334 PREMATURE END OF EXPRESSION NEAR OPERAND COLUMN 9

39 IF0334 PREMATURE END OF EXPRESSION NEAR OPERAND COLUMN 5

NUMBER OF STATEMENTS FLAGGED IN THIS ASSEMBLY = 4

HIGHEST SEVERITY WAS B

عناوين المتأثرات يجب أن تُصَفَّ حسب الحدود المناسبة . أما تلك التي تعالج السمات فلا يوجد أية مشكلة بالنسبة لها . إنَّ عدم المحافظة على هذه القواعد يؤدي إلى حدوث مشكلة في المؤول (أنظر المثل السطر 33) ، فهو يؤدي عند التنفيذ إلى انقطاع من نوع «Specification» (تميز) . التعليقات يجب أيضاً أن تُصَفَّ في حدود نصف كلمات .

تمارين

تمرين 1.9 - للتعليقات أدناه :

- 1 - إفحص إذا كانت العناصر التي تؤلف المتأثرات هي مُطلقة أو مُحوَّلة .
- 2 - باعتماد النسق المرتبطة بكل تعليمة نستخلص ، فقط حسب المعايير النحوية ، إذا كانت التعليقات صحيحة .
- 3 - قم بإجراء تأويل التعليقات الصحيحة .

		مرصف القاعدة = 12	
		النسق	
ADBASE	CSECT	*,12	
	USING	B,D	RX
	L	3,D(3)	RX
	LR	A,D	RR
	ST	D,X'4'(3,C)	RX
	L	A,B'1011'(3)	RX
	L	D,E(B)	RX
	L	A,E(B)	RX
	MVC	A(B,C),D	SS
	MVC	E(L'D),D	SS
	L	2,D+L'D	RX
	A	EQU 0	
	B	EQU 1	
	C	EQU 10	
	D	DS 5F	
	E	DS 12F	
END			

10 . التعليمات بلغة المؤول عموميات

سنقوم بدراسة التعليمات - الآلية حسب نوع التمثيل الداخلي الذي تُعالجه هذه التعليمات . من البديهي أن تكون التعليمات الحسابية العشرية ، مثلاً ، بدون معنى إلا عندما تُقدّم لها معطيات مؤكّدة عشرياً . مثلاً ، من الواضح أن المرافف المبلوغة بالتعليمات المتحركة هي مرافف متحركة .

سنبدأ بالتعليمات التي تعمل على المرافف العامة ، ولكن في البداية يجب عرض الترميز المعتمد .

1.10 . الترميز

سيتم تحديد التعليمات - الآلية حسب النسق التالي :

المعنى	كود العمليات سادس عشري	النسق	العوامل	كود - العملية الرمزي
LOAD (S ₂) → R ₁	COP=58	RX	R ₁ , D ₂ (X ₂ , B ₂)	L

تشير العوامل إلى العناوين مع قاعدة محدّدة بشكل جلي . أما الشروحات فتذكر هذا العنوان بشكل رمزي . فإذا S₂ سبّغي العنوان المحسوب بإضافة مضمون المرافف القاعدية والمؤشر إلى الإزاحة ، في المجموع فإن S₂=D₂+B₂+X₂ بالنسبة للتعليمات RX و S₂=D₂+B₂ للباقية .

سنجد في الحيز مُعاملات أو في الشروحات الرموز التالية :

- R₁, R₂ هي عبارة عن أرقام المرافف التي يمكن أن تُستبدل بالتعبير المطلقة .
- D قيمة الإزاحة بالنسبة إلى العنوان القاعدي .
- X رقم المرافف المؤشر المُستعمل .
- B رقم المرافف القاعدي .
- M قناع من أربع بتات موجود في التعليمات .

I قيمة فورية موجودة في التعليمات .
CO عدّاد البرنامج (Program counter) .
S عنوان رمزي ، تعبير قابل للتحويل :

$$S = D_2 + X_2 + B_2$$

$$S = D_2 + B_2 \text{ أو}$$

(S) مضمون العنوان S .
→ رمز للتخصيص ، أي نسخ منطقة في أخرى دون تهديم المنطقة الأصلية . مثلاً :
(S) → R₁ يعني نسخ مضمون المرصف R₁ في المنطقة من الذاكرة بالعنوان S . لن
نستعمل أبداً الترميز (R) للإشارة إلى مضمون المرصف R لأنه لن يوجد أي
إبهام ، في حالة المرصف يتعلق ذلك دائماً بالمضمون بينما يجب التمييز بين الاسم
S للذاكرة ومضمونها .

((S)) من الممكن استعمال هذا التعبير للإشارة إلى أن مضمون العنوان S هو نفسه المعتمد
كعنوان نأخذ منه المضمون .

CC يعني كود - الشرط .
الدلائل (indices) الدلائل 1 ، 2 ، 3 تُرجع إلى الحقول المرتبطة بالتعليمات الآلية (فقرة
1.5) .

R₁ (24-31) تعني البتات 24 إلى 31 من المرصف رقم R₁ .
R₁ ، R₁₊₁ تعني المرصف المزدوج المؤلف من المراصف ذات الرقم R₁ و R₁₊₁ . R₁ يكون
رقماً مزدوجاً .

العناوين (addresses) تشير إلى أن العناوين تعني البايته من اليسار لمنطقة ما ، وإن البتات
من الكلمة ، من مرصف ... هي مرقمة من اليسار إلى اليمين إنطلاقاً
من 0 .

(370) تشير إلى أن التعليمات غير موجودة إلا على المكنة 370 :

2.10 . كود العمليات الحرفية التذكيرية

كتابة كود - العمليات الرمزية يخضع إلى قواعد من المفيد الإشارة لها هنا . إن كود
العملية يترجم الفعل المطلوب إجراؤه . السمة الأولى (أحياناً السمتان الأوليان) هي
بداية الفعل الذي يُعبّر عن العمل .

مثلاً :

A	Add	جمع
L	LOAD	شحن

ST	STore	خزن
MVC	MoVe	نقل

الأحرف التالية هي معدلات (1) أو أنها تُمَيِّز نوع المعطيات المُعالِجَة (2) أو أيضاً النسق RR أو SI للتعلييات (3) .

أمثلة :

(1)	AL	جمع منطقي Add Logical
(2)	CVB	تحويل إلى ثنائي ConVert Character
(2)	AE	جمع معطيات من نوع بفاصلة متحركة قصير Add données de type E (flottant court)
(2)	MVC	نقل السّيات MoVe Characters
(2)	AD	جمع معطيات من نوع D
(3)	LR	شحن بنسق RR
(3)	LPR	شحن إيجابي بنسق RR
(3)	MVI	شحن مباشر بنسق SI

11 . الحساب بفاصلة ثابتة والحركات

- 1.11 . تعليمات الشحن والتخزين في المرافف العامة
هذه هي التعليمات التي تنسخ المتأثر في أحد المرافف :
« عنوان المتأثر ، رقم المرافف LOAD »
وتنسخ مضمون المرافف في الذاكرة على عنوان معين :
« عنوان ، رقم المرافف STORE »

هذه العمليات لا تؤثر على المتأثر الأساسي . بعض التعليمات تؤدي إلى تركيز كود
- الشرط CC ، لموقعين ثنائيين ينتميان إلى PSW (فصل 4) ، تبعاً لإشارة المتأثر المنقول
حسب الاتفاق التالي :

- بعد العملية فإن CC سيُرَكِّز على ⁽¹⁾ .
- 0 إذا كانت النتيجة صفراً .
 - 1 إذا كانت النتيجة سلبية .
 - 2 إذا كانت النتيجة إيجابية .
 - 3 إذا كان هناك زيادة عن السعة (overflow) .
- الزيادة عن السعة تؤدي عادة إلى إنقطاع في تنفيذ البرنامج . أي أنه سيحدث
خطأ يُعالجه نظام التشغيل . يوجد برنامج ، يُدعى برنامج إنقطاع «fixed point
overflow» ، يعطي العلاج للمستعمل ويوقف العمل في تنفيذ البرنامج بنهاية غير
طبيعية . بإمكان المبرمج أن يقوم بتقنيّة عملية الإنقطاع هذه في بعض الحالات بتركيز
البتات المناسبة لقناع البرنامج في PSW .
وسندرس هذا الأمر لاحقاً (الفصل 19) .

(1) هذا الاتفاق هو صالح فقط للتعليمتين LOAD وSTORE وبعض التعليمات الأخرى . وسنرى كيف يتم
تركيز CC لكل مجموعة تعليمات .

المتأثر 1 هو دائماً مرصف ، والمتأثر الثاني يُمكن أن يكون مرصفاً ، نصف كلمة أو كلمة - ذاكرة .

من المهم أن نشير إلى أن المتأثرات الموجودة على العناوين المشار إليها بواسطة S يجب أن تُحصر في حدود كلمات أو نصف - كلمات حسب التعليقات .

LR	R_1, R_2	RR	COP=18	LOAD
				$R_2 \rightarrow R_1$
L	$R_1, D_2(X_2, B_2)$	RX	COP=58	LOAD
				$(S_2) \rightarrow R_1$
			CC	لا يتغير

LH	$R_1, D_2(X_2, B_2)$	RX	COP=48	LOAD HALFWORD
				$(S_2) \rightarrow R_1$

يُعتبر المتأثر الثاني كعدد صحيح بإشارة وبطول 16 بتة . يُوسّع إلى 32 بتة قبل التحويل. CC لا يتأثر .

LCR	R_1, R_2	RR	COP=13	LOAD COMPLEMENT
				$R_2 \rightarrow R_1$

يُحزّن عكس (مكمّل إلى 2) في R_1 في R_2 . overflow إذا أكملنا العدد السليبي الأقصى . يوضع CC حسب الإشارة النهائية لـ R_1 .

LPR	R_1, R_2	RR	COP=10	LOAD POSITIVE
				$R_2 \rightarrow R_1$ القيمة المطلقة لـ

سيحدث زيادة عن السعة (overflow) إذا أكملنا العدد السليبي الأقصى . يُركّز CC على 0 ، 2 أو 3 حسب النتيجة .

LNR	R_1, R_2	RR	COP=11	LOAD NEGATIVE
-----	------------	----	--------	---------------

المكمّل إلى 2 للقيمة المطلقة لـ R_2 يُحزّن في R_1 . لن يحدث overflow . CC يُركّز على 0 أو 1 .

LTR	R_1, R_2	RR	COP=12	LOAD AND TEST
				$R_2 \rightarrow R_1$

تعليمية شبيهة بـ LR باستثناء كون الإشارة النهائية لـ R_1 تُركّز CC . R_1 يمكن أن يكون معادلاً لـ R_2 .

LM	$R_1, R_3, D_2(B_2)$	RS	COP=98	LOAD MULTIPLE
----	----------------------	----	--------	---------------

المواقع المتتالية للذاكرة ، انطلاقاً من العنوان S_2 مستشحن في المرصف العامة R_1 ، R_{1+1} ، R_3 ، ... في هذه التعليمة يُفترض بأن يتبع المرصف 0 المرصف 15 . هكذا :

LM 15, 1, ALPHA
 مستشحن الكلمة ذات العنوان ALPHA في المرصف 15 ، وتلك ذات العنوان ALPHA+4 في المرصف 0 وهكذا دواليك . تُستخدم هذه التعليمة بشكل خاص لترميم إطار البرنامج .
 CC لا يتأثر .

LA $R_1, D_2(X_2, B_2)$ RX COP=41 LOAD ADDRESS

$S_2 \rightarrow R_1(8-31)$ $0 \rightarrow R_1(0-7)$

تُخزَّن القيمة ذات العنوان S_2 في البتات من 8 إلى 31 من المرصف R_1 .
يتم تصفير البتات من 0 إلى 7. وتنطبق هنا قواعد حساب العنوان،
أي أن القيمة $D_2 + X_2 + B_2$ تُخزَّن (عنوان فعلي). من الممكن أن
نأخذ نفس المرصف لـ R_1 ، X_2 أو B_2 . المرصف 0 لا يؤخذ أبداً وكأنه
قاعدة أو مرصف تأثير.

الاستعمال: أنظر التمارين

- شحن عنوان في مرصف،

- شحن عدد غير سلمي أصغر أو يعادل 4095 (القيمة القصوى للإزاحة)
في مرصف،

- زيادة مضمون مرصف بقيم أصغر أو تساوي 4095.

IC $R_1, D_2(X_2, B_2)$ RX COP=43 INSERT CHARACTER

$(S_2) \rightarrow R_1(24-31)$

لا يتغيَّر $R_1(0-23)$

يتم تخزين بايت واحدة بعنوان S_2 في R_1 . CC لا يتأثر.

ICM $R_1, M_3, D_2(B_2)$ RS COP=BF INSERT CHARACTERS UNDER MASK (370)

تُرَبط البتات الأربع من القناع M_3 بالبتات الأربع للمرصف R_1 .
البايتات من R_1 المرتبطة بالبتات «1» من القناع يتم شحنها مع البايتات
المتتالية من S_2 . طول المتأثر الثاني يعادل عدد «1» في القناع.
يُركَّز كود الشرط:

CC = 0 : جميع البتات الداخلة هي مصفَّرة أو القناع مصفَّر،

CC = 1 : البتة ذات الوزن الأكبر في S_2 هي «1»،

CC = 2 : البتات ذات الوزن الأكبر في S_2 هي «0» ولكن جميع البتات
الداخلة ليست صفراً.

وفي الختام فإن CC يُركَّز حسب إشارة S_2 .

ST $R_1, D_2(X_2, B_2)$ RX COP=50 STORE
 $R_1 \rightarrow (S_2)$

CC والمرصف R_1 يبقيان بدون تعديل.

STH $R_1, D_2(X_2, B_2)$ RX COP=40 STORE HALFWORD
 $R_1(16-31) \rightarrow (S_2)$

المتأثر الثاني هو بطول 2 بايت. CC يبقى بدون تعديل.

STM $R_1, R_3, D_2(B_2)$ RS COP=90 STORE MULTIPLE

المراسف العامة من R_1 إلى R_3 يتم تخزينها في مواقع متتالية من الذاكرة
بدءاً من العنوان S_2 . الرقم 0 للمرصف 0 مُفترض أنه يتبع الرقم 15
بشكل يؤدي معه تنفيذ التعليمة ALPHA 15, 1, ST إلى تخزين

المرافف 15 ، 0 ، 1 بالعناوين ALPHA ، ALPHA+4 ، ...
تستخدم التعليمة بشكل خاص لحفظ إطار البرنامج . CC يبقى بدون
تغيير .

STC $R_1, D_2(X_2, B_2)$ RX COP=42 STORE CHARACTER
 $R_1(24-31) \rightarrow (S_2)$

R_1 و CC يبقيان بدون تعديل .

STCM $R_1, M_3, D_2(B_2)$ RS COP=BE STORE CHARACTERS UNDER MASK
(370)

البتات الأربع من القناع M_3 ترتبط بالأربع بايتات من المرصف R_1 .
أما بايتات R_1 ، والمختارة بوجود «1» في القناع ، فيتم تخزينها بشكل
متراص على العنوان S_2 . كود الشرط CC لا يتغير .

2.11 . التعليقات الحسابية بفاصلة ثابتة

هي التعليقات التي تعمل على معطيات ممثلة بفاصلة ثابتة . تكوّد القيم السلبية
بواسطة المكمل إلى 2 . كما تقوم بالعمليات الأربع الأساسية بين مرصف ومرصف أو
بين مرصف وذاكرة . الضرب والجمع يستعملان مرافف مزدوجة (فقرة 1.10) . هذه
التعليقات تؤدي إلى تعديل CC حسب إشارة النتيجة ، وحسب الإتفاق الجاري كما في
1.11 .

CC = 0 إذا كانت النتيجة صفراً .

CC = 1 إذا كانت النتيجة سلبية .

CC = 2 إذا كانت النتيجة إيجابية .

CC = 3 إذا كان هناك overflow .

يمكن قطع التعليمة في حالة حدوث حادثة غير طبيعية ، كما يلي :

- عنوان من خارج المنطقة المخصصة .
- جبهة متأثر غير صحيحة ، مرصف مزدوج معني بشكل سيء .
- فيض عن السعة overflow .

AR R_1, R_2 RR COP=1A ADD
 $R_1 + R_2 \rightarrow R_1$
A $R_1, D_2(X_2, B_2)$ RX COP=5A ADD
 $R_1 + (S_2) \rightarrow R_1$

لا يتغير المتأثر الثاني . يتم تركيز كود الشرط CC ، احتمال
حصول overflow .

AH $R_1, D_2(X_2, B_2)$ RX COP=4A ADD HALFWORD
 $(S_2) + R_1 \rightarrow R_1$

المتأثر (S_2) هو على نصف كلمة . يُوسَّع إلى كلمة قبل العملية . يتم تركيز CC . احتمال حصول Overflow .

SR R_1, R_2 RR COP=1B SUBTRACT
 $R_1 - R_2 \rightarrow R_1$
 S $R_1, D_2(X_2, B_2)$ RX COP=5B SUBTRACT
 $R_1 - (S_2) \rightarrow R_1$

المتأثر الثاني لا يتعدل . يتم تركيز CC .

SH $R_1, D_2(X_2, B_2)$ RX COP=4B SUBTRACT HALFWORD
 $R_1 - (S_2) \rightarrow R_1$

المتأثر S_2 هو على نصف كلمة ، يُوسَّع إلى 32 بتة قبل العملية . يتم تركيز CC .

MR R_1, R_2 RR COP=1C MULTIPLY
 $R_{1+1} \times R_2 \rightarrow R_1, R_{1+1}$
 M $R_1, D_2(X_2, B_2)$ RX COP=5C MULTIPLY
 $R_{1+1} \times (S_2) \rightarrow R_1, R_{1+1}$

المرصف R_1 المذكور في التعليمة يجب أن يكون مرصفاً مزدوجاً . المتأثر الأول يجب أن يكون موجوداً في R_{1+1} ومحصوراً لجهة الشمال . النتيجة ستوضع في R_1 ، R_{1+1} . لا احتمال لحدوث overflow ، لا يتم تركيز CC .

MH $R_1, D_2(X_2, B_2)$ RX COP=4C MULTIPLY HALFWORD
 $R_{1+1} \times (S_2) \rightarrow R_1, R_{1+1}$

المرصف R_1 يجب أن يكون مرصفاً مزدوجاً ، S_2 يتألف من 16 بتة ويُعتبر كعدد صحيح بإشارة يُوسَّع إلى 32 بتة قبل العملية . لا يحدث overflow ولا يتم تركيز CC .

DR R_1, R_2 RR COP=1D DIVIDE
 $R_1, R_{1+1} : R_2 \begin{matrix} \rightarrow R_1 \text{ باقي} \\ \rightarrow R_{1+1} \text{ قيمة القسمة} \end{matrix}$

R_1 هو مرصف مزدوج . يتمتع الباقي بنفس إشارة المقسوم . عندما لا تسع 32 بتة نتيجة القسمة يحدث overflow . لا يتم تركيز CC .

D $R_1, D_2(X_2, B_2)$ RX COP=5D DIVIDE
 $R_1, R_{1+1} : (S_2) \begin{matrix} \rightarrow R_1 \text{ reste} \\ \rightarrow R_{1+1} \text{ quotient} \end{matrix}$

R_1 يجب أن يكون مرصفاً مزدوجاً . للباقي نفس إشارة المقسوم . عندما لا تسع 32 بتة نتيجة القسمة يكون هناك فيض عن السعة . لا يتم تركيز CC .

ملاحظات :

دراسة هذه التعليقات تسمح لنا بملاحظة إن النتيجة تحل دائماً مكان المتأثر الأول الذي يضيع منا . بينما لا يتم تعديل المتأثر الثاني . التعليقات التي تجري على نصف كلمة تفترض توسيع نصف الكلمة إلى كلمة قبل العملية .

3.11 . عمليات المقارنة بفاصلة ثابتة

تؤثر تعليمات المقارنة فقط على مضمون كود الشرط . هذه التعليقات هي خاصة حسب نوع تمثيل المعطيات المقارنة . سندرس هنا تلك المتعلقة بالفاصلة الثابتة . كما في التعليقات التي رأيناها ، فإن المتأثر الأول هو دائماً موجود في مرصف معين والمتأثر الثاني في مرصف آخر أو في الذاكرة . يجري تركيز CC حسب الطريقة التالية :

CC = 0 إذا كان المتأثر الأول = المتأثر الثاني .

CC = 1 إذا كان المتأثر الأول أصغر من المتأثر الثاني .

CC = 2 إذا كان المتأثر الأول أكبر من المتأثر الثاني .

CC = 3 لا يستعمل .

CR R₁,R₂ RR COP=19 COMPARE

C R₁,D₂(X₂,B₂) RX COP=59 COMPARE

المقارنة هي جبرية وتتعلق بـ 32 بته . يتم تركيز مضمون CC .

CH R₁,D₂(X₂,B₂) RX COP=49 COMPARE HALFWORD

يوسع المتأثر الثاني إلى 32 بته قبل المقارنة مع إنتشار بته الإشارة . يتم تركيز CC .

4.11 الجمع والطرح المنطقي

نعني بالجمع والطرح المنطقي ، تعليقات تعدل مضمون CC بطريقة مختلفة عن الجمع والطرح العادي الذي رأيناه أعلاه . إضافة لذلك فإن overflow لا يؤدي إلى قطع البرنامج

يتم تركيز CC على الشكل التالي :

CC = 0 إذا كانت النتيجة صفراً بدون مرّحل .

CC = 1 إذا كانت النتيجة مختلفة عن 0 بدون مرّحل (no carry)

CC = 2 إذا كانت النتيجة صفراً مع مرّحل .

CC = 3 إذا كانت النتيجة مختلفة عن صفر مع مرّحل .

ALR R_1, R_2 RR COP=1E ADD LOGICAL
 $R_2 + R_1 \rightarrow R_1$

AL $R_1, D_2(X_2, B_2)$ RX COP=5E ADD LOGICAL
 $(S_2) + R_1 \rightarrow R_1$

SLR R_1, R_2 RR COP=1F SUBTRACT LOGICAL
 $R_1 - R_2 \rightarrow R_1$

SL $R_1, D_2(X_2, B_2)$ RX COP=5F SUBTRACT LOGICAL
 $R_1 - (S_2) \rightarrow R_1$

5.11 . التحريك من الذاكرة إلى الذاكرة

تتم في أغلب الأحيان بواسطة تعليمات من نوع SS . لا يوجد أي تقييد فيما يتعلق بالاصطفاف (alignment) . يمكن أن يتم تركيز الطول بشكل واضح في التعليمات : MVC ZONE 1, ZONE 2 أو ضمناً MVC ZONE 1 (L), ZONE 2 . يقوم عندها المؤول باختيار خاصية - الطول الخاصة بالتأثير الأول L'ZONE 1 . الطول المؤول هو الطول المذكور في التعليمات ناقص 1 . يمكن للمتأثرين أن يتراكبا ، ونجد هذه الميزة مستعملة في التمرين 6.11 .

MVI $D_1(B_1), I_2$ SI COP=92 MOVE
 $I_2 \rightarrow (S_1)$

يتم تخزين البايته المباشرة I_2 في S_1 .

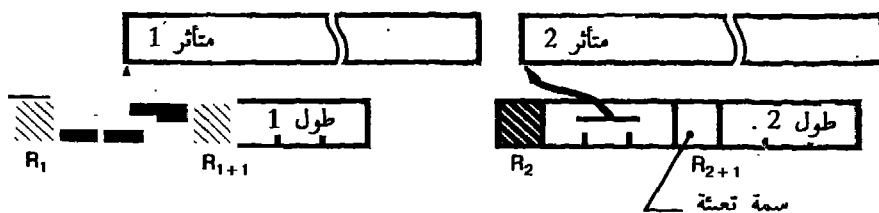
MVC $D_1(L, B_1), D_2(B_2)$ SS COP=D2 MOVE

$(S_1) \rightarrow (S_2)$ بطول L .

الحركة تتم من اليسار إلى اليمين . العملية هي غير قابلة للانقطاع عند نقل بايتين . يسمح بالترابك وفي هذه الحالة يجدر الانتباه إلى أن الحركة تجري من اليسار إلى اليمين من أجل الحصول على النتيجة .

MVCL R_1, R_2 RR COP=0E MOVE LONG
(370)

نسخ التأثير الثاني في التأثير الأول .
 R_1 (8 - 31) يحتوي على عنوان التأثير الأول ،
 $R_1 + 1$ (8 - 31) طول التأثير الأول ،
 R_2 (8 - 31) عنوان التأثير الثاني ،
 $R_2 + 1$ (0 - 7) سمة تعبئة ،
 $R_2 + 8$ (8 - 31) طول التأثير الثاني .



الحركة تتم من اليسار إلى اليمين ، لكل بايتة على حدة . التعليمة هي قابلة للانقطاع عند نسخ بايتين . إذا كان طول المتأثر الثاني هو أصغر من طول المتأثر الأول ، يتم تكملة المتأثر الأول بسمة تعبئة . يمكن تراكب المناطق بشرط أن لا يقوم النسخ بتعديل بايتة جرى تعديلها سابقاً .

يجري تركيز CC على الشكل التالي :

CC = 0 إذا كان كلا المتأثرين بنفس الطول ،

CC = 1 المتأثر الأول هو أقصر ،

CC = 2 المتأثر الأول هو أطول ،

CC = 3 إذا أدت عملية التطابق إلى تعديل في بايتة معدلة أصلاً .

يمكن إستعمال هذه التعليمة لتصفير الذاكرة .

MVN $D_1(L, B_1), D_2(B_2)$ SS COP=D1 MOVE NUMERIC

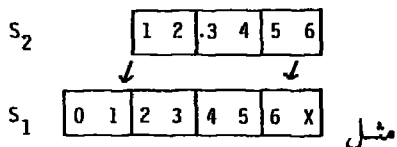
نسخ نصف - بايتات بالوزن الأضعف من (S2) في أنصاف - بايتات الوزن الأضعف من (S1) . تبقى أنصاف - البايتات بالوزن الأقوى دون تعديل . يسمح بالتراكب وبهذا الصدد نعطي الملاحظة نفسها كما بالنسبة لـ MVC .

MVZ $D_1(L, B_1), D_2(B_2)$ SS COP=D3 MOVE ZONES

نسخ نصف بايتات بالوزن الأقوى من (S2) في نصف بايتات الوزن الأقوى من (S1) . تبقى أنصاف - البايتات بالوزن الأضعف دون تعديل . يسمح بتراكب الحيزات وبهذا الصدد نعطي الملاحظة نفسها كما بالنسبة لـ MVC .

MVO $D_1(L_1, B_1), D_2(L_2, B_2)$ SS COP=F1 MOVE WITH OFFSET

نسخ من (S2) في (S1) مع إزاحة إلى اليسار مقدار نصف بايتة . العملية تتم من اليمين إلى اليسار ، بايتة بعد بايتة . لا يتم تغيير آخر بايتة لجهة اليمين .



تمارين

تمرين 1.11 - ضع في الصفر الثنائي أحد المرافف (أعط حلين لتعليلة واحدة دون حجز ثوابت) .

تمرين 2.11 - غير إشارة المرصف (تمثيل ثنائي) .

تمرين 3.11 - ضع جميع بتات المرصف في 1 .

تمرين 4.11 - اشحن القيمة 2048 في مرصف ، ثم القيمة 4095 (دون حجز ثابتة) بعد ذلك اشحن 4096 .

تمرين 5.11 - زد مضمون أحد المرافف مقدار 4 .

تمرين 6.11 - عبيء منطقة بطول $L \geq 256$ بايتة بنجوم (تعليماتان) .

12 التفريعات

نفهم بالتفريع كل تعديل في مضمون عداد البرنامج يؤدي إلى إنقطاع في الدوران المتتالي للتعلييات .

عودتنا دراسة اللغات المتطورة على اعتبار نوعين من الإنقطاعات في المتتالية :

- الإنقطاعات الإلزامية (GOTO في لغة فورتران) .

- الإنقطاعات المشروطة (IF) .

في لغة المؤول ، فإن الإنقطاعات المشروطة تنتج إما عن اختيار لقيمة مأخوذة من كود الشرط ، إما عن اختبار لقيمة مأخوذة من مرصف عام . التعليمتان BC و BCR تفحصان كود الشرط CC والتعلييات BCT ، BCTR ، BXH ، BXLE تُخفّض أو تزيد من مضمون مرصف وبعد ذلك تفحص قيمته .

يمكن تنفيذ الإنقطاعات الإلزامية بواسطة BC و BCR .

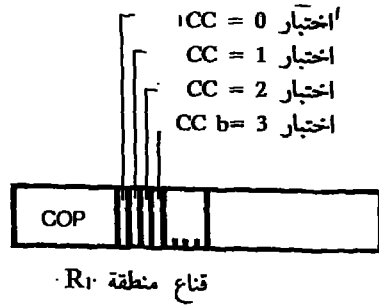
1.12 . الكود - الشرط

لقد التقيناه عند دراسة التعلييات السابقة . ونذكر بأنه عبارة عن مؤشر بموقعين ثنائيين ، ينتميان إلى PSW (الببتان، 34 ، 35) ويركّزان بواسطة بضعب تعلييات حسب النتيجة الحاصلة . التعلييات الحسابية ، مثلاً ، التركيز حسب إشارة النتيجة ، تعلييات المقارنة حسب القيمة النسبية لتأثرين .

الكود الشرطي CC يمكن أن يأخذ إذن أربع قيم ثنائية 00 ، 01 ، 10 ، 11 يتم مراجعتها في التعلييات بواسطة 0 ، 1 ، 2 ، 3 .

2.12 . التعلييات التي تفحص الكود الشرطي (CC) : BCR و BC

هذه التعلييات تستعمل المنطقة R₁ ، المكوّنة من أربع بتات ثنائية ، من نسقها الآلي ، ليس كرقم مرصف بل كقناع : كل بة تعادل 1. وموجودة في هذه المنطقة تناسب اختبار إحدى القيم الأربع التي نحصل عليها بواسطة CC حسب الإتفاق التالي :



هكذا ، فالقناع المُعادل 1100 (ثنائياً) سيسمح باختبار الشروط $CC=0$ أو $CC=1$. الشرط المُختار فعلاً يتعلّق إذاً بالتعلّمة التي أدت إلى تركيز CC .
لقد رأينا أن CC تركّز حسب الطريقة التالية :

كود الشرط	0	1	2	3
تعلّمات حسابية نتيجة	=0	<0	>0	فيض عن السعة
تعلّمات مقارنة متأثر أول	=	<2°	>2°	---

القناع المُعادل لـ 1100 (أي C بالنظام السادس عشري أو 12 بالعشري) يناسب الاختبارات التالية :

- نتيجة سلبية أو صفر بعد تعلّمة حسابية .
- متأثر أول أصغر من المتأثر الثاني بعد تعلّمة مقارنة .

BCR M_1, R_2 RR COP=07 BRANCH ON CONDITION

M_1 هي القناع المذكور أعلاه .

بعد تنفيذ الشرط ، هناك تفريع إلى العنوان المخزّن في R_2 . ولا سيتابع التنفيذ بالتوالي . مما يترجم على الشكل التالي : الشرط المنفذ $R_2 \rightarrow CO$
ولا $CO + 2 \rightarrow CO$

BC $M_1, D_2(X_2, B_2)$ RX COP=47 BRANCH ON CONDITION

M_1 قناع .

إذا تم تنفيذ الشرط فسيحدث تفريع إلى العنوان $D_2 + X_2 + B_2$
وإلا فإن التنفيذ سيتتابع بالتوالي ، مما يترجم على الشكل التالي :

في حال تنفيذ الشرط : $D_2 + X_2 + B_2 \rightarrow CO$

وإلا $CO + 4 \rightarrow CO$

$D_2 + X_2 + B_2$ عنوان التفريع ..

في لغة المؤول ، يُحدّد القناع M_1 بواسطة تعبير مطلق ، عادة رقم عشري .
BCR 15,R أو BC 15, ALPHA يناسبان القناع 1111 . يتعلّق ذلك إذا
بالتفريع المنتظم لأنه مهما تكن قيمة CC هناك تفريع .
BCR 0,R أو BC 0,ALPHA هي عبارة عن تعليقات دون فعل لأنه لن يتم
إختبار أي شرط . وهي تميّز بأنها بدون فعل .
الأكواد الحرفية التذكيرية الموسّعة
وفي النهاية كي يتم تفادي تحديد القناع الخاص ولتذكر الإتفاقات المذكورة أعلاه ،
فإن المؤول يسمح باستعمال كود حرفي حسب الشرط المفحوص .
ويقوم بمهمة ترجمة الكود الحرفي إلى BC أو BCR .

هكذا :

يناسب تفريعاً غير شرطي $B \quad D_2(X_2, B_2)$
BC 15, $D_2(X_2, B_2)$

يناسب تفريعاً غير شرطي $BR \quad R_2$
BCR 15, R_2

يناسب تفريعاً معيّناً وإلا يعادل $BNE \quad D_2(X_2, B_2)$
BC 7, $D_2(X_2, B_2)$

سنجد في الملحق اللائحة الكاملة للكود الحرفي التذكيري الموسّع . سنلاحظ إن
الأكواد الحرفية تتعلّق بالتعليمة التي تقوم بتركيز الكود الشرطي . من المفيد ، لوضوح
البرنامج ، إستعمال هذه الأكواد الحرفية التذكيرية . ونركّز على كون هذه الأكواد العملية
لا تتناسب سوى مع 2 كود - مكنة . ونشير ، كما ذكرنا في الفقرة 2.10 ، إلى أن الأكواد
التي تنتهي بـ R تناسب تعليقات بنسق RR أو BCR .

3.12 . . التعليقات التي تفحص القيمة المأخوذة من مرصف (مؤشر)
أربع تعليقات BCT ، BCTR ، BXH و BXLE تسمح بتعديل مضمون

المرصف والتفرع إلى عنوان معين عندما تصبح قيمته معادلة ، أقل أو أكبر من كمية محدّدة .

BCTR R_1, R_2 RR COP=06 BRANCH ON COUNT
 $R_1 - 1 \rightarrow R_1$

إذا كانت $R_1 \neq 0$: $R_2 \rightarrow CO$ (تفرع إلى العنوان الموجود في R_2) .
 وإلا : $CO + 2 \rightarrow CO$ (تنفيذ التعليمة التالية) .
 ملاحظة : إذا كان R_2 هو المرصف 0 فالعدّ يتم بدون تفرع .

BCT $R_1, D_2(X_2, B_2)$ RX COP=46 BRANCH ON COUNT
 $R_1 - 1 \rightarrow F_1$

إذا : $R_1 \neq 0$: $S_2 \rightarrow CO$ (تفرع إلى العنوان S_2)
 وإلا : $CO + 4 \rightarrow CO$ (تنفيذ التعليمة التالية) .

BXH $R_1, R_3, D_2(B_2)$ RS COP=86 BRANCH ON INDEX HIGH

1- زيادة مضمون R_1 : $R_1 + R_3 \rightarrow R_1$
 2- عندما تصبح R_1 أكبر من المرجعية : تفرع . المرجعية هي R_3

R_{3+1}

1- R_3 هو مرصف برقم مفرد .

R_3 هو مرجع المقارنة والزيادة .

فإذا : $R_1 + R_3 \rightarrow R_1$ بعد ذلك ، إذا كان $R_1 > R_3$

عندئذ $S_2 \rightarrow CO$ (تفرع إلى S_2)

وإلا : $CO + 4 \rightarrow CO$ (متابعة على التوالي) .

ب- R_3 هو مرصف برقم مزدوج

نستعمل المرصف المزدوج R_3 و R_{3+1}

R_3 هو الزيادة و R_{3+1} هو المرجعية . إذن $R_1 + R_3 \rightarrow R_1$ ثم

إذا كان $R_1 > R_{3+1}$ عندئذ $S_2 \rightarrow CO$ (التفرع إلى S_2) .

وإلا : $CO + 4 \rightarrow CO$ (متابعة المتتالية) .

ملاحظة :

يجب أن لا نخلط هنا بين المصطلح إشارة مع مرصف المؤشر

للتعليقات RX .

المقارنة تتم جبرياً . ويتم إهمال overflow عند الجمع .

BXLE $R_1, R_3, D_2(B_2)$ RS COP=87 BRANCH ON INDEX LOW OR EQUAL

1- زيادة R_1 : $R_1 + R_3 \rightarrow R_1$

2- عندما يصبح R_1 أصغر أو يعادل المرجعية : تفريع المرجعية . المرجعية
 * هي R_3 أو R_{3+1} .

أ- R_3 هو مرصف برقم مفرد .

R_3 هو مرجعية المقارنة والزيادة .

فإذا : $R_1 \rightarrow R_1 + R_3$ بعد ذلك ، إذا كان $R_1 \leq R_3$ عندئذ
 $CO \rightarrow S_2$ (تفريع إلى S_2) وإلا $CO \rightarrow CO + 4$ (متابع)
 المتتالية)

ب- R_3 هو مرصف برقم مزدوج .

R_3 هو الزيادة ، R_{3+1} هو المرجعية .

فإذا : $R_1 \rightarrow R_1 + R_3$ ثم إذا كان $R_1 \leq R_{3+1}$ عندئذ $CO \rightarrow S_2$
 (تفريع إلى S_2) وإلا $CO \rightarrow CO + 4$ (متابع المتتالية) .

ملاحظة : يجب أن لا نخلط هنا بين المصطلح مؤشر مع مرصف المؤشر
 للعمليات RX . تتم المقارنة جبرياً . يتم إهمال overflow عند الجمع .

4.12 . تفريع مع عودة

مشكلة التفريع مع تخزين عنوان التعليمة التي تلي تعليمة التفريع تحدث عند دعوة
 برنامج ثانوي . هناك تعليمتان BALR و BALR موجهتان لهذا الإستعمال .

BALR R_1, R_2 RR COP=05 BRANCH AND LINK
 CO $\rightarrow R_1(8-31)$ (تخزين عنوان العودة)
 CC $\rightarrow R_1(0-7)$
 $R_2(8-31) \rightarrow CO$ (تفريع)

ملاحظة :

ندكر بأن قيمة عداد البرنامج CO تتغير خلال تنفيذ التعليمة .
 هكذا ، فعنوان التعليمة التالية حسب BALR هو المخزن في R_1 .
 BALR $R_1, 0$ يقوم بتخزين العنوان التالي في R_1 ولكن لا تفريع . هناك
 إذن تنابع للمتتالية . هذا الشكل هو الأكثر استعمالاً لشحن مرصف
 قاعدي بالقيمة التالية لعداد البرنامج .

إذا كانت التعليمة BALR موجودة على العنوان 50000 ، فإن
 القيمة 50002 ستخزن في R_1 .

BAL $R_1, D_2(X_2, B_2)$ RX COP=45 BRANCH AND LINK
 CO $\rightarrow R_1(8-31)$ (تخزين عنوان العودة)
 CC $\rightarrow R_1(0-7)$
 $S_2 \rightarrow CO$ (تفريع إلى العنوان S_2)

كما في BALR ، فعنوان التعليمة التالية سيخزن في R_1 . إذا كانت
 BAL موجودة على العنوان 50000 فإن مضمون R_1 هو 50004 .

EX R₁,D₂(X₂,B₂) RX COP=44 EXECUTE

هذه التعليمة تسمح بتنفيذ تعليمة واحدة موجودة خارج التابع الطبيعي للعنوان S₂ . بعد ذلك ، فإن العمل يُعاود بالتوالي .

يتم تنفيذ عملية « أو » متضمنة بين البتات R₁(24-31) و S₂(8-15) تسمح بتعديل هذا الحقل من التعليمة (رقم المرصف ، قيمة تلقائية أو طول) . إذا كان R₁ هو المرصف 0 فلا يتم تنفيذ العملية « أو » (OR) . كما لا يمكن تنفيذ عملية التحويل .

تطبيق

عندما نرغب بإجراء نقل للمعلومات MVC من منطقة لا نعرف طولها إلا في لحظة التنبؤ - هذه الحالة تحدث عند معالجة التسجيلات بطول متغير ، يكون طول الفقرة موجوداً في رأسها - من الممكن إذاً تنفيذ التعليمة «MVC» . والطريقة هي التالية : شحن الطول في R₁(24-31) :

(تنقيص 1)
BCTR R₁, 0
EX R₁,MOVE
-- -----
MOVE MVC -----

تُنفَّذ MVC مع الطول المطلوب دون أن يكون هناك تعديل للتعليمة في الذاكرة . التعليمة MVC لا تتعدّل إلا خلال مدة التنفيذ . ويمكن أن تكون موجودة في أي مكان ولكن يُفضّل أن تكون EX و MVC موجودتين في نفس الصفحة من الذاكرة كي لا تقع في خطأ محتمل في نقص الصفحة .

تمارين :

تمرين 1.12 . أكتب متتالية التعليقات التي تسمح بتكرار N مرة إحدى عمليات المعالجة .
تمرين 2.12 . إحسب مجموع عناصر جدول من الكلمات يحتوي على أعداد بفاصلة ثابتة .

تمرين 3.12 . إعكس سلسلة من السيات CH1 في CH2 .
تمرين 4.12 . نقّص مضمون المرصف 1 (تعليمة واحدة) .
تمرين 5.12 . إشحن مرصفاً معيناً بالعنوان الجاري زائد 2 .

13 . العمليات المنطقية

1.13 . الدوال المنطقية

يسمح الكومبيوتر IBM 360/370 بعنوان البايته ، ومن غير الممكن الإشارة إلى بته معيّنة داخل البايته . ولكن بسبب وجود تعليمات الإزاحة (Shift) والتعليمات المنطقية سيكون بإمكاننا إختبار أو تعديل مضمون إحدى البتات من داخل الكلمة .

العمليات المنطقية الموجودة هي « و » (AND) ، الجمع « أو » (OR) و « أو المقتصرة » (EOR) . جدول العمليات المنطقية هو التالي :

A	1	0	1	0	تعليمات
B	1	1	0	0	
A AND B	1	0	0	0	NR N NI NC
A OR B	1	1	1	0	OR O OI OC
A FOR B	0	1	1	0	XR X XI XC

2.13 . التعليمات المنطقية

المتأثرات هي :

- مرصفان عامان (شكل RR) : التعليمات NR ، OR ، XR ،
 - مرصف وكلمة - ذاكرة (شكل RX) : التعليمات N ، O ، X ،
 - بايته موجودة في التعليمة وبايته موجودة في الذاكرة (الشكل SI عنوانه مباشرة) : التعليمات NI ، OI ، XI ،
 - سلسلتان من البايتات في الذاكرة (شكل SS) : التعليمات NC ، OC ، IC .
- توضع النتيجة دائماً في المتأثر 1.
- يتم تركيز كود الشرط حسب الطريقة التالية :

CC	
0	إذا كانت النتيجة تعادل صفر
1	إذا كانت النتيجة مختلفة عن صفر

عمليات الإنقطاع الممكنة تتعلّق ، كالعادة ، بمسألة العنوان : تعدّ على المنطقة المخصّصة من الذاكرة ، تعدّ على المنطقة الممكنة من الذاكرة أو مشكلة الزيادة في مضمون المرافف المزدوجة .

التقاطع « و » (AND)

NR R_1, R_2 RR COP=14 AND
 $R_1 \text{ «AND» } R_2 \rightarrow R_1$
 تتم العملية على 4 بايتات .

N $R_1, D_2(X_2, B_2)$ RX COP=54 AND
 $R_1 \text{ «AND» } (S_2) \rightarrow R_1$
 تتم العملية على أربع بايتات .

NI $D_1(B_1), I_2$ SI COP=94 AND
 $(S_1) \text{ «AND» } I_2 \rightarrow (S_1)$
 I_2 هي قيمة تلقائية موجودة في التعليميّة . العملية تتم على بايتة واحدة .

NC $D_1(L, B_1), D_2(B_2)$ SS COP=D4 AND
 $(S_1) \text{ «AND» } (S_2) \rightarrow (S_1)$

العملية تتم بين منطقتين من الذاكرة بطول مشترك هو L بايتة . وتجري العملية بايتة بعد بايتة من اليسار إلى اليمين . كل شيء يسير كما لو كانت كل بايتة محسوبة وتخزّنة في الذاكرة قبل العبور إلى البايّة التالية .

تطبيق عملي :

تصفير إحدى البتات .

الجمع «أو»

OR R_1, R_2 RR COP=16 OR
 $R_1 \text{ «OR» } R_2 \rightarrow R_1$
 تتم على أربع بايتات

O $R_1, D_2(X_2, B_2)$ RX COP=56 OR
 $(S_2) \text{ «OR» } R_1 \rightarrow R_1$
 تتم على أربع بايتات .

OI $D_1(B_1), I_2$ SI COP=96 OR
(S₁) «OU» I₂ → (S₁)

I₂ هي قيمة موجودة في التعليم . تجري العملية على بايتة واحدة .

OC $D_1(L, B_1), D_2(B_2)$ SS COP=D6 OR
(S₂) «OR» (S₁) → (S₁)

تتم العملية على منطقتين من الذاكرة بطول مشترك هو L بايتة . ويتم بايتة بعد أخرى من اليسار إلى اليمين .

تطبيق عملي :

جعل إحدى البتات تعادل 1 .

«أو المقتصرة» (EOR)

XR R_1, R_2 RR COP=17 EXCLUSIVE OR
 $R_1 \oplus \text{«EOR»} R_2 \rightarrow R_1$

تتم العملية على أربع بايتات .

X $R_1, D_2(X_2, B_2)$ RX COP=57 EXCLUSIVE OR
 $R_1 \oplus \text{«EOR»} (S_2) \rightarrow R_1$

تتم العملية على أربع بايتات .

XI $D_1(B_1), I_2$ SI COP=97 EXCLUSIVE OR
(S₁) «EOR» I₂ → (S₁)

I₂ هي قيمة تلقائية موجودة في التعليم . تتم العملية على بايتة واحدة .

XC $D_1(L, B_1), D_2(B_2)$ SS COP=D7 EXCLUSIVE OR
(S₁) «EOR» (S₂) → (S₁)

تجري العملية على منطقتين من الذاكرة بطول مشترك L بايتة ، وتجري بايتة بعد بايتة من اليسار إلى اليمين كما لو كانت كل بايتة قد جرى حسابها وتخزينها في الذاكرة قبل العبور إلى البايته التالية .

تطبيق عملي :

عكس البتة ، مكمل منطقي ، تصفير منطقة من الذاكرة .

3.13 . المقارنات المنطقية

كما في جميع العمليات المنطقية تجري معالجة جميع البتات بنفس الطريقة . لا وجود لأي تمييز للبتة ذات الوزن الأعلى . تتم المقارنة من اليسار إلى اليمين وتتوقف عند أول معادلة . يُركّز كود الشرط حسب الطريقة التالية :

(نذكر أنّ المتأثر الأول هو ذلك الذي يتم بلوغه في التعليم بواسطة المؤشر 1 .

الإنقطاعات الممكنة هي تلك المتعلقة بالعنونة وتلك المتعلقة بحدود الكلمات .

0	إذا كانت التأثيرات متساوية
1	إذا كان التأثير الأول أصغر من التأثير الثاني
2	إذا كان التأثير الأول أكبر من التأثير الثاني
3	غير مستعمل

CLR R_1, R_2 RR COP=15 COMPARE LOGICAL

مقارنة بين كامل المرافف .

CL $R_1, D_2(X_2, B_2)$ RX COP=55 COMPARE LOGICAL

مقارنة على أربع بايتات .

CLI $D_1(B_1), I_2$ SI COP=95 COMPARE LOGICAL

مقارنة منطقية مباغرة بين القيمة I_2 الموجودة في التعليمة و (S_1) .

CLC $D_1(L, B_1), D_2(B_2)$ SS COP=D5 COMPARE LOGICAL

مقارنة بين سلاسل تمتد حتى 256 بايتة بطول مشترك L .

CLM $R_1, M_3, D_2(B_2)$ RS COP=BD COMPARE LOGICAL CHARACTERS UNDER MASK (370)

القناع M_3 ، المكوّن من أربع بتات يختار في R_1 من 0 إلى 4 بايتات تُقارن بالبايتات المتتالية إنطلاقاً من العنوان S_2 . البتة الأولى من القناع ، إذا كانت معادلة لـ 1 تختار البايته الأولى من R_1 وهكذا دواليك . يتم تركيز CC .

القناع المعادل لـ 1011 يختار البايتات 0 ، 2 ، 3 من R_1 التي تتم مقارنتها مع ثلاث بايتات إنطلاقاً من S_2 . المقارنة تتم من اليسار إلى اليمين .

CLCL R_1, R_2 RR COP=OF COMPARE LOGICAL LONG (370)

مقارنة بين سلسلتين من البايتات حيث العناوين والأطوال موجودة في المرافف المزدوجة حسب الإتفاق التالي :



تجري العملية من اليسار إلى اليمين من خلال العناوين 1 و 2 . إذا لم يكن طول السلسلتين متعادلاً ، يُفترض بأن يُكْمَل الأقصر من اليمين بالسمة «padding» (سمة الحشو) .

العملية تتم بايئة بعد بايئة مع زيادة عناوين وتقصير الطول . وهي قابلة للإيقاف بين مقارنة باييتين . وتتوقف عند أول لا معادلة نلتقيها أو في نهاية السلسلة مع تركيز كود - الشرط .

4.13 . مقارنات منطقية خاصة

لقد قمنا هنا بتصنيف التعليقات التي ، زيادة عن وظيفتها في المقارنة ، تتمتع بعمل خاص . هذه التعليقات تركز كود الشرط بصورة مختلفة .

CS $R_1, R_3, D_2(B_2)$ RS COP=BA COMPARE AND SWAP
(370) مقارنة بين R_1 و (S_2)

إذا : $R_1 = (S_2)$ عندئذ $R_3 \rightarrow (S_2)$ و $CC \rightarrow 0$.
إذا : $R_1 \neq (S_2)$ عندئذ $R_1 \rightarrow (S_2)$ و $CC \rightarrow 1$.

CDS $R_1, R_3, D_2(B_2)$ RS COP=BB COMPARE DOUBLE AND SWAP
(370) مقارنة بين R_1 و (S_2)

إذا : $R_1 = (S_2)$ عندئذ $R_3 \rightarrow (S_2)$ و $CC \rightarrow 0$.
إذا : $R_1 \neq (S_2)$ عندئذ $R_1 \rightarrow (S_2)$ و $CC \rightarrow 1$.

المقارنة CDS تتم على 64 بتة . وبالنسبة فإن R_3 و R_1 هما مرصفتان مزدوجتان (فقرة 1.10) و S_2 هو عنوان كلمة مزدوجة من الذاكرة .

تُستعمل هاتين التعليقتين لتنفيذ المزامنة بين مهمتين تقسمان منطقة مشتركة من الذاكرة . عندما تتم المعادلة ، فإن كل بلوغ للعنوان S_2 هو ممنوع لأي مُعالج مركزي حتى نهاية عملية النقل $(S_2) \rightarrow R_3$.

TM $D_1(B_1), I_2$ SI COP=91 TEST UNDER MASK

TM تقوم باختبار حالة البتات من البايئة ذات العنوان S_1 . I_2 هي قناع من 8 بتات ، كل «1» ، موجود في القناع يسمح باختبار وجود بتة «1» في الموقع المناسب من البايئة S_1 .

مثلاً : القناع $X'60$ أي $B'01100000$ يفحص وجود «1» في الموقعين 1 و 2 من البايئة . ويجري إهمال المواقع الأخرى . وفي الإجمال ، فإن TM يقوم بتنفيذ عملية AND منطقية بين البايئة التي تم فحصها والقناع دون تعديل البايئة ولكن بتركيز كود الشرط فقط :

$CC = 0$: جميع البتات التي جرى اختبارها هي 0 أو القناع هو في صفر ،
 $CC = 1$: بعض البتات هي صفر ، وأخرى هي 1 ،
 $CC = 2$: غير مستعمل
 $CC = 3$: جميع البتات المختبرة هي 1 .

11001110	11001110	11001110	البايتة المختبرة
00110000	11001000	01011100	القناع
--00----	11--1----	-1-011--	AND
0	3	1	CC

تطبيق :

TM يبدو وكأنه ينتمي إلى CLI . وفعلاً فإن TM يُعتمد لاختبار البتات أكثر من البايتات . مثلاً ، لمعرفة ما إذا كانت البايتة هي رقمية نستعمل CLI لأن القيمة يجب أن تكون محصورة بين F0 و F9 .

TM يمكن أن تُستعمل لتنفيذ تأثير متعدد .

تمارين :

نذكر أن الدالة «AND» تسمح بجعل البتات تعادل صفراً ، وإن الدالة «OR» تسمح بجعلها 1 وإن «EOR» تسمح بعكسها .

تمرين 1.13 . ضع في صفر ثنائي منطقة بطول $L \geq 256$ بايتة ، مرصفاً ، بايتة .

تمرين 2.13 . اكتب التعليمة التي تسمح بتركيز قيمة كود الطول في تعليمة من نوع SS .

تمرين 3.13 . بدّل مضمون منطقتين من الذاكرة ، مرصفين ، ربعين من البتات من نفس البايتة .

تمرين 4.13 . تعرّف ما إذا كانت منطقة من الذاكرة مملوءة بفراغ أو بصفر ثنائي

تمرين 5.13 . قم بإجراء تأثير يؤدي إلى تفريع مرّة على اثنتين بواسطة تحويل منطقة قناع تعليمة BC 15,... إلى BC 0,... .

تمرين 6.13 . قم بإجراء تأثير يؤدي إلى تفريع إلى جميع نقاط العبور ما عدا الأول .

تمرين 7.13 . بدّل جميع أصفار الينار (X'F0') في عدد عشري بفراغات (X'40') .

تمرين 8.13 . البايتة تسمح بتجميع حتى ثمانية مؤشرات ثنائية . لنأخذ البايتة INDIC التي تجمع المؤشرات الثنائية INDLECR ، INDECR ، INDWAIT ، المناسبة على التوالي للقيم السادس عشرية X'80' ، X'40' و X'20' من INDIC (تحتل المؤشرات البتات 0 ، 1 و 2 من INDIC) . اكتب التعليقات التي تسمح :

- بتعريف INDIC ، INDLECR ، INDECR ، INDWAIT ؛
- بتركيز INDWAIT في 1 ؛
- بتركيز INDLECR و INDWAIT في 1 ؛
- بتركيز INDLECR و INDECR في صفر ؛

- بتفريع إلى ALPHA إذا كانت INDWAIT في «1» ؛
 - بتفريع إلى BETA إذا كانت INDWAIT و INDLEC في «1» ؛
 - بتفريع إلى GAMMA عندما يكون فقط INDLEC أو INDWAIT في «1» ؛
 - بتفريع إلى DELTA عندما تكون INDWAIT و INDLEC في صفر .
- لنفترض بأننا نرغب بربط INDLEC بالبتة 7 من INDIC بدلاً من البتة 0 ، مما يتناسب مع X'01 بدلاً من X'80 . الحل الخاص بكم هل يسمح بعدم تعديل تعليقات التركيز والاختبار لـ INDLEC ؟

14 . عمليات الإزاحة (Shift)

1.14 . التعليقات « المنطقية » والتعليقات « الحسابية »

عند دراسة تعليقات الجمع بفاصلة ثابتة ، لاحظنا ، أنه الى جانب التعليقات A ، AR و AH ، تأتي عمليات الجمع المنطقية . الفرق بين هذين النوعين من العمليات هو التالي :

- تميّز العمليات الجبرية البتة 0 ، المعتبرة كإشارة ، تجري العملية على 31 بتة مع مُرحّل محتمل إلى بتة الإشارة . يجري اختيار الإشارة ويمكن أن تؤدي إلى إنقطاع من نوع overflow .

- العمليات من نوع منطقي لا تأخذ بعين الاعتبار أي تمييز للبتة ذات الوزن الأكبر . تجري معالجة جميع البتات بنفس الطريقة . أي ترحيل في نهاية البتة ذات الوزن الأكبر لا يؤدي إلى انقطاع .

الإزاحة هي عبارة عن نقل إلى اليسار أو إلى اليمين لعدد n من المواقع لتشكيلة ثنائية موجودة في مرصف بسيط (إزاحة بسيطة) أو في مرصف مزدوج (إزاحة مزدوجة) .

عند الإزاحة تضيع البتات المطرودة . والبتات الداخلة لجهة اليمين هي دائماً صفر . أما البتات التي تدخل من اليسار فيمكن أن تكون إما «0» (إزاحة منطقية إلى اليمين أو إزاحة حسابية إلى اليمين لعدد إيجابي) أو «1» (إزاحة جبرية إلى اليمين لعدد سلمي) . سنرى السبب لاحقاً .

2.14 . الإزاحة الجبرية

تجري الإزاحة الجبرية على القيمة ، أي على 31 بتة (إزاحة بسيطة) أو على 63 بتة (إزاحة مزدوجة) .

- الإزاحة إلى اليمين تؤدي إلى إدخال بتات معادلة لبتة الإشارة .

- الإزاحة إلى اليسار تؤدي إلى إدخال 0. إذا جرى تعديل بته الإشارة سيحدث إنقطاع من نوع overflow بفاصلة ثابتة .

الإزاحة الجبرية تؤدي إلى تركيز كود الشرط على الشكل التالي :

CC = 0	إذا كانت النتيجة صفراً .
CC = 1	إذا كانت النتيجة سالبة .
CC = 2	إذا كانت النتيجة إيجابية
CC = 3	إذا كان يوجد overflow (تعديل في بته الإشارة في حالة إزاحة إلى اليسار) .

أمثلة :

لتبسيط العرض سنفترض إن حجم المصرف يعادل ثمان بتات . البته ذات الوزن الأكبر هي إذا بته الإشارة .

مرصف بسيط	مرصف مزدوج
<p>قبل الإزاحة</p> $\begin{array}{ c } \hline 00001111 \\ \hline S \end{array} = +15$	$\begin{array}{ c c } \hline 00000000 & 11001111 \\ \hline S \end{array} = +207$
<p>بعد الإزاحة لجهة اليسار ثلاثة</p> $\begin{array}{ c } \hline 01111000 \\ \hline S \end{array} = +120$ <p>CC = 2</p>	$\begin{array}{ c c } \hline 00000110 & 01111000 \\ \hline S \end{array} = +1656$
<p>بعد الإزاحة لجهة اليمين ثلاثة</p> $\begin{array}{ c } \hline 00000001 \\ \hline S \end{array} = +1$ <p>CC = 2</p>	$\begin{array}{ c c } \hline 00000000 & 00011001 \\ \hline S \end{array} = +25$
<hr/>	
<p>قبل الإزاحة</p> $\begin{array}{ c } \hline 11100101 \\ \hline S \end{array} = -27$	
<p>بعد الإزاحة لجهة اليمين 1</p> $\begin{array}{ c } \hline 11110010 \\ \hline S \end{array} = -14$ <p>CC = 1</p>	
<p>بعد الإزاحة اليسار 4</p> $\begin{array}{ c } \hline 01010000 \\ \hline S \end{array} = +80$ <p>CC = 3 OVERFLOW</p>	

3.14 . الإزاحة المنطقية

تعالج الإزاحة المنطقية 32 بتة (إزاحة بسيطة) أو 64 بتة (إزاحة مزدوجة) دون أخذ بالاعتبار البتة ذات الوزن الأكبر . البتات الداخلة هي دائماً «0» . لا يحدث إنقطاع من نوع overflow . لا يجري تعديل في CC .
أمثلة : على ثمان بتات .

10011100

قبل الإزاحة

01110000

بعد الإزاحة لجهة اليسار 2

00100111

بعد الإزاحة لجهة اليمين 2

4.14 . تعليمات الإزاحة

يوجد أربع عمليات إزاحة جبرية ، أربع تعليمات إزاحة منطقية ، وتعليمات إزاحة لعدد عشري . سنرى هذه الأخيرة عند دراسة الحساب العشري .
الإزاحة الجبرية :

SLA	$R_1, D_2(B_2)$	RS	COP=8B	SHIFT LEFT SINGLE	إزاحة بسيطة إلى اليسار
SLDA	$R_1, D_2(B_2)$	RS	COP=8F	SHIFT LEFT DOUBLE	إزاحة مزدوجة إلى اليمين
SRA	$R_1, D_2(B_2)$	RS	COP=8A	SHIFT RIGHT SINGLE	إزاحة بسيطة إلى اليمين
SRDA	$R_1, D_2(B_2)$	RS	COP=8E	SHIFT RIGHT DOUBLE	إزاحة مزدوجة إلى اليمين

الإزاحة المنطقية

SLL	$R_1, D_2(B_2)$	RS	COP=89	SHIFT LEFT SINGLE LOGICAL	إزاحة بسيطة منطقية إلى اليسار
SLDL	$R_1, D_2(B_2)$	RS	COP=8D	SHIFT LEFT DOUBLE LOGICAL	إزاحة منطقية مزدوجة إلى اليسار

SRL $R_1, D_2(B_2)$ RS COP=88 SHIFT RIGHT SINGLE LOGICAL
إزاحة بسيطة منطقية إلى اليمين

SRDL $R_1, D_2(B_2)$ RS COP=8C SHIFT RIGHT DOUBLE LOGICAL
إزاحة مزدوجة منطقية إلى اليمين

قواعد مشتركة للإزاحات المنطقية والجبرية

- تتم عمليات الإزاحة على مضمون المرصف R_1 .
- بالنسبة لعمليات الإزاحة المزدوجة ، فإن R_1 يجب أن يكون مرصفاً مزدوجاً حسب الاتفاق العادي (فقرة 2.10).
- المتأثر الثاني $D_2(B_2)$ ليس عنواناً :
- 1- إذا كان B_2 هو المرصف 0 ، فإن البتات الست ذات الوزن الأضعف للنقطة تعطي عدد المواقع المطلوب لإزاحتها . $SLA 5,3$ أو $SLA 5,3(0)$ هما عمليتان إزاحة لجهة اليسار لثلاثة مواقع ثنائية .
- 2- إذا لم يكن B_2 هو المرصف 0 ، فإن المرصف المذكور يحتوي على عدد المواقع المطلوب لإزاحتها . ونحصل على الإزاحة بشكل غير مباشر . $SRDL 6,0(5)$ يزحل منطقياً المرصف المزدوج (المرصفان 6 و 7) لعدد المواقع المشار إليها في المرصف 5 .
- وحدها عمليات الإزاحة الجبرية تقوم بتركيز كود الشرط CC حسب اتفاق الفقرة 2.14 .

تمارين :

- تمرين 1.14 - ضع في صفر مرصفاً بواسطة الإزاحة .
- تمرين 2.14 - اضرب واقسم عدداً موجوداً في مرصف على قوة 2 بواسطة الإزاحة . إفحص ، بالنسبة للقسمة ، اتجاه التقريب .
- تمرين 3.14 - إفحص فيما إذا كان زوج من المراصف مزدوج / مفرد هو صفر .
- تمرين 4.14 - برمجة إزاحة دائرية لمرصف بسيط .

15 . مسائل

1.15 . الفرز

يتعلق ذلك بترتيب جدول من الكلمات التي تحتوي على أعداد بفاصلة ثابتة بترتيب تصاعدي . لقد قمنا باختيار الخوارزم الكلاسيكي الذي يُعرف بـ « طريقة الفقاعة » . تقوم الطريقة على فحص عناصر الجدول من اليسار إلى اليمين مع تبديل العناصر المتتالية الموجودة بشكل عشوائي . نضع إلى اليمين العنصر الأكبر كما نلاحظ من المثل التالي :

5	1	3	2
1	5	3	2
1	3	5	2
1	3	2	5

إذا كان N هو حجم الجدول ، نبدأ العملية باعتماد الجدول الثانوي بالحجم $N-1$ وهكذا دواليك ، طالما يوجد عملية تبديل واحدة على الأقل خلال التكرار السابق .

ولو افترضنا أنه خلال فحص الأعداد ، لم تجر أية عملية تبديل فمعنى ذلك إن الترتيب قد حصل .

البرنامج مؤلف من حلقتين BCL1 و BCL2 متداخلتين . الحلقة الداخلية BCL2 تفحص الجدول باستعمال مرصيف مؤشر PTR : (PTR) هو عنوان العنصر . العناصر التي جرت مقارنتها هي إذاً ((PTR)) و ((PTR)+4)⁽¹⁾ . يتم إنشاء الحلقة بواسطة BXLE . المرصيف المزدوج INCRE/REFER يحتوي على الزيادة 4 والحلقة $TAB+(N-1) * 4$.

عند إجراء تبديل نقوم بتركيز البايته INDIC في 1 . الحلقة BCL1 تُكرر BCL2 طالما إن $INDIC=1$.

(1) نذكر بأنه حسب الترميز المعتمد ، (PTR) يُقرأ « مضمون PTR » وهنا هو إذن عبارة عن عنوان . مضمون هذا العنوان ، أي العنصر المطلوب ، يُرمز إليه بـ ((PTR)) .

قبل الفرز

PSW AT ENTRY TO SNAP 078D1000 0008705E ILC 2 INTC 0033

REGS AT ENTRY TO SNAP

FLTR 0-6 0000000000000000 003000000000000000 000000000000000000 000000000000000000
REGS 0-7 00001A0 80087038 80084F64 00087010 00000004 000870F8 00084F58 000000FF
REGS 8-15 00000000 00084EB0 00084F00 00087000 00087010 000870FC 0008C758 00000000

-STORAGE

087000 4510C018 8F087148 0A134140 00044150 90ECD00C 18CF5000 C0F041D0 C0EC0700 *0.....
087020 00087148 00000000 00087050 00000000 C0E80700 4510C048 00720000 0000A400 *Y.....
087040 C1344780 C0829400 00087050 00000000 C0087010 80087146 9680101C 0A339500 *X.....
087060 00045072 00049601 C1344120 C0C41B54 58720000 59720004 47D0C07A D2032000 *A.....D.....
087080 20045072 00049601 C1348724 C06047F0 C04E0700 4510C0A8 00720000 0000A400 *A.....0.....
0870A0 00087148 00000000 00087050 00000000 C0087010 00087146 9680101C 0A330700 *0.....
0870C0 4510C088 80087148 0A1458D0 C0F098EC D00C07FE 0000000F FFF0FFFD 00000002 *0.....
0870E0 00000007 00000008 FFF0FFFE 00000002 00000009 00000000 00000000 *0.....
087100 00087F58 00000000 00000000 00000000 00000000 00000000 00000000 *0.....
087120 0008A080 00000000 00000000 00000000 00000000 00000000 00000000 *0.....
087140 00000000 01000000 00000000 00000000 00000000 00000000 00000000 *0.....

بعد الفرز

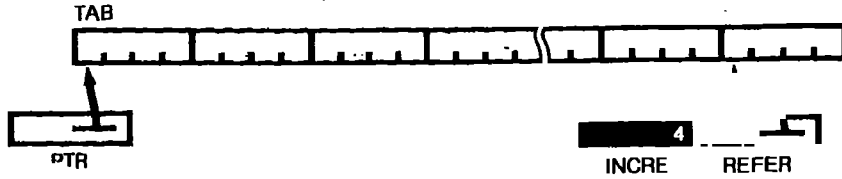
PSW AT ENTRY TO SNAP 078D1000 0008705E ILC 2 INTC 0033

REGS AT ENTRY TO SNAP

FLTR 0-6 0000000000000000 000000000000000000 000000000000000000 000000000000000000
REGS 0-7 008DC698 80087098 000870E8 00087010 00000004 000870E4 00084F58 00000002
REGS 8-15 00000000 00084EB0 00084F00 00087000 00087010 000870FC 0008C758 00000000

-STORAGE

087000 4510C018 8F087148 0A134140 00044150 90ECD00C 18CF5000 C0F041D0 C0EC0700 *0.....
087020 00087148 00000000 00087050 00000000 C0E80700 4510C048 00720000 0000A400 *Y.....
087040 C1344780 C0829400 00087050 00000000 C0087010 80087146 9680101C 0A339500 *X.....
087060 00045072 00049601 C1344120 C0C41B54 58720000 59720004 47D0C07A D2032000 *A.....D.....
087080 20045072 00049601 C1348724 C06047F0 C04E0700 4510C0A8 00720000 0000A400 *A.....0.....
0870A0 00087148 00000000 00087050 00000000 C0087010 00087146 9680101C 0A330700 *0.....
0870C0 4510C088 80087148 0A1458D0 C0F098EC D00C07FE 0000000F FFF0FFFD 00000002 *0.....
0870E0 00000007 00000008 FFF0FFFE 00000002 00000009 00000000 00000000 *0.....
087100 00087F58 00000000 00000000 00000000 00000000 00000000 00000000 *0.....
087120 0008A080 00000000 00000000 00000000 00000000 00000000 00000000 *0.....
087140 00000000 01000000 00000000 00000000 00000000 00000000 00000000 *0.....



من الممكن أن نكتب الخوارزم على الشكل التالي :

(← هورمز التخصيص)
 INCRE ← 4
 INDIC ← 1
 REFER ← TAB+(N-1)*4
 BCL1: TANT QUE INDIC ≠ 0 FAIRE
 INDIC ← 0
 REFER ← REFER - INCRE
 BCL2: -----
 exploration

 FIN BCL2
 FIN BCL1

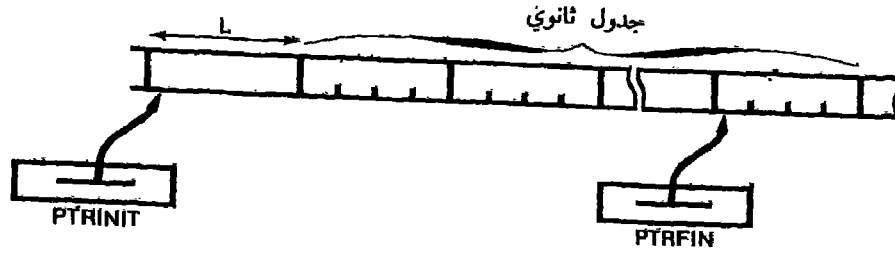
SNAP هي عبارة عن ماكرو تعليمية نموذجية تسمح بالحصول على صورة سادس عشرية من الذاكرة . إستعمالها يتطلب فتح السجل (OPEN) ، إغلاق (CLOSE) ووصف السجل بواسطة الماكرو تعليمية PRINT NOGEN.DCB (سطر 2) تسمح بإلغاء توليد كود الماكرو تعليمات .

2.15 . إستشارة فرقانية للجدول

يقوم البرنامج على البحث عن وجود أو غياب معلومة من داخل أحد الجداول . البحث المتسلسل يبدو صعباً ويستهلك كثيراً من الوقت عندما يصبح حجم الجدول كبيراً . من الممكن أن نستعمل طريقة الفرقان عندما تكون العناصر منظمة . والصيغة هي التالية :

لنفترض جدولاً TAB من N عنصر منظم نبحث فيه عن موقع المعلومة الموجودة في MOT . نقوم باستشارة العنصر الموجود في وسط TAB ونقارنه بـ MOT . البحث ينتهي عندما نجد التعادل . وإلا نُعيد الكرة ونتابع الاستشارة باختيار واحد من الجدولين الثانويين المشكّلين بواسطة القسمة السابقة حسب موقع العنصر الذي نبحث عنه بالنسبة للعنصر الوسط . بعد كل إستشارة تضيق الفسحة التي نبحث فيها إلى النصف .

سنفترض إن طول العنصر هو L وهذا الطول يعادل قوة (أس) P للعدد 2 ($L=2^p$) . هذا سيسمح بإجراء عمليات ضرب وقسمة بواسطة الإزاحة . سنستعمل مرادف مؤشرات لبلوغ العناصر . PTRINIT سيحتوي على عنوان العنصر الأول من الجدول الثانوي ناقص PTRFIN.L سيحتوي على عنوان العنصر الأخير من الجدول الثانوي .



عدد العناصر هو إذاً : $\frac{PTRFIN - PTRINIT}{L}$

عنوان العنصر الوسط هو :

عنوان البداية + $\frac{1}{2} \times$ عدد العناصر $\times L$

أي : $PTRINIT + L + \frac{1}{2} \left(\frac{PTRFIN - PTRINIT}{L} \right) \times L$

عند القسمة على L يجب إهمال الباقي الذي قد يظهر .

البرنامج التالي جرى اختباره بعد إجراء نداء لبرنامجين ثانويين مكتوبين بلغة فورتران : LIRE و ECR . وجود نداءات بلغة فورتران من خلال برنامج رئيسي بلغة المؤول يتطلب كتابة التعليقات 59 و 60 غير الموجودة إذن إلا لأسباب توافقية بإشراف النظام المستعمل (FORTRAN G, OS-VS2) .

LDC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000000				37	DICHO START 0
000001	18CF	0000C	00000	38	* PIRINT EQU 5
000002	5C0D	0000C	00000	39	PIRINT EQU 2
000003	18B0	0000C	00000	40	PIRFIN EQU 3
000004	4100	0000C	00000	41	PIRELEM EQU 4
000005	50D2	0000C	00000	42	WORK EQU 5
000006	4770	0000C	00000	43	WORK EQU 6
000007	4770	0000C	00000	44	WORK EQU 7
000008	4770	0000C	00000	45	WORK EQU 8
000009	4770	0000C	00000	46	WORK EQU 9
000010	4770	0000C	00000	47	WORK EQU 10
000011	4770	0000C	00000	48	WORK EQU 11
000012	4770	0000C	00000	49	WORK EQU 12
000013	4770	0000C	00000	50	WORK EQU 13
000014	4770	0000C	00000	51	WORK EQU 14
000015	4770	0000C	00000	52	WORK EQU 15
000016	4770	0000C	00000	53	WORK EQU 16
000017	4770	0000C	00000	54	WORK EQU 17
000018	4770	0000C	00000	55	WORK EQU 18
000019	4770	0000C	00000	56	WORK EQU 19
000020	4770	0000C	00000	57	WORK EQU 20
000021	4770	0000C	00000	58	WORK EQU 21
000022	4770	0000C	00000	59	WORK EQU 22
000023	4770	0000C	00000	60	WORK EQU 23
000024	4770	0000C	00000	61	WORK EQU 24
000025	4770	0000C	00000	62	WORK EQU 25
000026	4770	0000C	00000	63	WORK EQU 26
000027	4770	0000C	00000	64	WORK EQU 27
000028	4770	0000C	00000	65	WORK EQU 28
000029	4770	0000C	00000	66	WORK EQU 29
000030	4770	0000C	00000	67	WORK EQU 30
000031	4770	0000C	00000	68	WORK EQU 31
000032	4770	0000C	00000	69	WORK EQU 32
000033	4770	0000C	00000	70	WORK EQU 33
000034	4770	0000C	00000	71	WORK EQU 34
000035	4770	0000C	00000	72	WORK EQU 35
000036	4770	0000C	00000	73	WORK EQU 36
000037	4770	0000C	00000	74	WORK EQU 37
000038	4770	0000C	00000	75	WORK EQU 38
000039	4770	0000C	00000	76	WORK EQU 39
000040	4770	0000C	00000	77	WORK EQU 40
000041	4770	0000C	00000	78	WORK EQU 41
000042	4770	0000C	00000	79	WORK EQU 42
000043	4770	0000C	00000	80	WORK EQU 43
000044	4770	0000C	00000	81	WORK EQU 44
000045	4770	0000C	00000	82	WORK EQU 45
000046	4770	0000C	00000	83	WORK EQU 46
000047	4770	0000C	00000	84	WORK EQU 47
000048	4770	0000C	00000	85	WORK EQU 48
000049	4770	0000C	00000	86	WORK EQU 49
000050	4770	0000C	00000	87	WORK EQU 50
000051	4770	0000C	00000	88	WORK EQU 51
000052	4770	0000C	00000	89	WORK EQU 52
000053	4770	0000C	00000	90	WORK EQU 53
000054	4770	0000C	00000	91	WORK EQU 54
000055	4770	0000C	00000	92	WORK EQU 55
000056	4770	0000C	00000	93	WORK EQU 56
000057	4770	0000C	00000	94	WORK EQU 57
000058	4770	0000C	00000	95	WORK EQU 58
000059	4770	0000C	00000	96	WORK EQU 59
000060	4770	0000C	00000	97	WORK EQU 60
000061	4770	0000C	00000	98	WORK EQU 61
000062	4770	0000C	00000	99	WORK EQU 62
000063	4770	0000C	00000	100	WORK EQU 63
000064	4770	0000C	00000	101	WORK EQU 64
000065	4770	0000C	00000	102	WORK EQU 65
000066	4770	0000C	00000	103	WORK EQU 66
000067	4770	0000C	00000	104	WORK EQU 67
000068	4770	0000C	00000	105	WORK EQU 68
000069	4770	0000C	00000	106	WORK EQU 69
000070	4770	0000C	00000	107	WORK EQU 70
000071	4770	0000C	00000	108	WORK EQU 71
000072	4770	0000C	00000	109	WORK EQU 72
000073	4770	0000C	00000	110	WORK EQU 73
000074	4770	0000C	00000	111	WORK EQU 74
000075	4770	0000C	00000	112	WORK EQU 75
000076	4770	0000C	00000	113	WORK EQU 76
000077	4770	0000C	00000	114	WORK EQU 77
000078	4770	0000C	00000	115	WORK EQU 78
000079	4770	0000C	00000	116	WORK EQU 79
000080	4770	0000C	00000	117	WORK EQU 80
000081	4770	0000C	00000	118	WORK EQU 81
000082	4770	0000C	00000	119	WORK EQU 82
000083	4770	0000C	00000	120	WORK EQU 83
000084	4770	0000C	00000	121	WORK EQU 84
000085	4770	0000C	00000	122	WORK EQU 85
000086	4770	0000C	00000	123	WORK EQU 86
000087	4770	0000C	00000	124	WORK EQU 87
000088	4770	0000C	00000	125	WORK EQU 88
000089	4770	0000C	00000	126	WORK EQU 89
000090	4770	0000C	00000	127	WORK EQU 90
000091	4770	0000C	00000	128	WORK EQU 91
000092	4770	0000C	00000	129	WORK EQU 92
000093	4770	0000C	00000	130	WORK EQU 93
000094	4770	0000C	00000	131	WORK EQU 94
000095	4770	0000C	00000	132	WORK EQU 95
000096	4770	0000C	00000	133	WORK EQU 96
000097	4770	0000C	00000	134	WORK EQU 97
000098	4770	0000C	00000	135	WORK EQU 98
000099	4770	0000C	00000	136	WORK EQU 99
000100	4770	0000C	00000	137	WORK EQU 100
000101	4770	0000C	00000	138	WORK EQU 101
000102	4770	0000C	00000	139	WORK EQU 102
000103	4770	0000C	00000	140	WORK EQU 103
000104	4770	0000C	00000	141	WORK EQU 104
000105	4770	0000C	00000	142	WORK EQU 105
000106	4770	0000C	00000	143	WORK EQU 106
000107	4770	0000C	00000	144	WORK EQU 107
000108	4770	0000C	00000	145	WORK EQU 108
000109	4770	0000C	00000	146	WORK EQU 109
000110	4770	0000C	00000	147	WORK EQU 110
000111	4770	0000C	00000	148	WORK EQU 111
000112	4770	0000C	00000	149	WORK EQU 112
000113	4770	0000C	00000	150	WORK EQU 113
000114	4770	0000C	00000	151	WORK EQU 114
000115	4770	0000C	00000	152	WORK EQU 115
000116	4770	0000C	00000	153	WORK EQU 116
000117	4770	0000C	00000	154	WORK EQU 117
000118	4770	0000C	00000	155	WORK EQU 118
000119	4770	0000C	00000	156	WORK EQU 119
000120	4770	0000C	00000	157	WORK EQU 120
000121	4770	0000C	00000	158	WORK EQU 121
000122	4770	0000C	00000	159	WORK EQU 122
000123	4770	0000C	00000	160	WORK EQU 123
000124	4770	0000C	00000	161	WORK EQU 124
000125	4770	0000C	00000	162	WORK EQU 125
000126	4770	0000C	00000	163	WORK EQU 126
000127	4770	0000C	00000	164	WORK EQU 127
000128	4770	0000C	00000	165	WORK EQU 128
000129	4770	0000C	00000	166	WORK EQU 129
000130	4770	0000C	00000	167	WORK EQU 130
000131	4770	0000C	00000	168	WORK EQU 131
000132	4770	0000C	00000	169	WORK EQU 132
000133	4770	0000C	00000	170	WORK EQU 133
000134	4770	0000C	00000	171	WORK EQU 134
000135	4770	0000C	00000	172	WORK EQU 135
000136	4770	0000C	00000	173	WORK EQU 136
000137	4770	0000C	00000	174	WORK EQU 137
000138	4770	0000C	00000	175	WORK EQU 138
000139	4770	0000C	00000	176	WORK EQU 139
000140	4770	0000C	00000	177	WORK EQU 140
000141	4770	0000C	00000	178	WORK EQU 141
000142	4770	0000C	00000	179	WORK EQU 142
000143	4770	0000C	00000	180	WORK EQU 143
000144	4770	0000C	00000	181	WORK EQU 144
000145	4770	0000C	00000	182	WORK EQU 145
000146	4770	0000C	00000	183	WORK EQU 146
000147	4770	0000C	00000	184	WORK EQU 147
000148	4770	0000C	00000	185	WORK EQU 148
000149	4770	0000C	00000	186	WORK EQU 149
000150	4770	0000C	00000	187	WORK EQU 150
000151	4770	0000C	00000	188	WORK EQU 151
000152	4770	0000C	00000	189	WORK EQU 152
000153	4770	0000C	00000	190	WORK EQU 153
000154	4770	0000C	00000	191	WORK EQU 154
000155	4770	0000C	00000	192	WORK EQU 155
000156	4770	0000C	00000	193	WORK EQU 156
000157	4770	0000C	00000	194	WORK EQU 157
000158	4770	0000C	00000	195	WORK EQU 158
000159	4770	0000C	00000	196	WORK EQU 159
000160	4770	0000C	00000	197	WORK EQU 160
000161	4770	0000C	00000	198	WORK EQU 161
000162	4770	0000C	00000	199	WORK EQU 162
000163	4770	0000C	00000	200	WORK EQU 163
000164	4770	0000C	00000	201	WORK EQU 164
000165	4770	0000C	00000	202	WORK EQU 165
000166	4770	0000C	00000	203	WORK EQU 166
000167	4770	0000C	00000	204	WORK EQU 167
000168	4770	0000C	00000	205	WORK EQU 168
000169	4770	0000C	00000	206	WORK EQU 169
000170	4770	0000C	00000	207	WORK EQU 170
000171	4770	0000C	00000	208	WORK EQU 171
000172	4770	0000C	00000	209	WORK EQU 172
000173</					

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT	ELEM < (MOT)	ELEM > (MOT)
0000DE	1823			116 INF	DS OH		
0000DE	47F0 C0EA	000EA		117	PTINIT,PTRELEM		
0000E0				118	TESTFIN		
0000E4	1823			120 SUP	DS OH		
0000E4	47F0 C0EA	000EA		121	PTFIN,PTRELEM		
0000E6	5820 C128	00128		122	PTFIN,PTRELEM		
0000EA	1562			124 TESTFIN	DS OH		
0000EA	47F0 C0EA	000EA		125	PTINIT,PTRELEM		
0000EC	47F0 C0EA	000EA		126	PTINIT,PTRELEM		
0000F0	47F0 C0EA	000EA		127	PTINIT,PTRELEM		
0000F4	D703 C130 C130	00130		129 NONTRCUV	DS OH		
0000F4	47F0 C11A	0011A		130	RANG,RANG		
000116				131	ECR,(MOT,RANG)		
000116				144	EPILIQUE		
00011A	4680 C06C	0006C		147	EPILIQUE		
00011A	58D0 C01C	0001C		148	DS OH		
000122	98EC D00C	0000C		149	SELECT		
000126	07FE			150	13,SAVEAREA+4		
000128	C0000004	0000A		151	LM 14		
00012C				153	* ZONE DE DONNES		
000130				154	N EQU 10		
000134	C000C01000000002			155	LONG		
00013C	C000C01000000008			156	MOT		
000144	C000C01000000020			157	RANG		
00014C	C000C01000000080			158	TAB		
000154	C000C010000000200			159	END		
000160	C000C00000			160	DICHO		
00016C	C0000000			161	=F12,		
000164	C0000000						

NB DE MOTS DE LA TABLE
LONGUEUR D'UN ELEMENT

16 . الحساب العشري

1.16 . عموميات

تقدّم التعليقات الحسابية العشرية وسائل لإجراء الحسابات على الأعداد العشرية « المتراسة packed » التي رأيناها في الفقرة 3.5.2. ج. ولاحقاً سندرس عملية تحويلها لمعطيات .

التعليقات الحسابية هي بنسق SS وتستعمل الطولين L_1 و L_2 للمتأثرين . يبقى طول المتأثرات محدوداً بـ 16 بايتة (31 رقماً عشرياً زائد الإشارة في التمثيل المتراس و 16 رقماً وإشارة في التمثيل الموسّع) لأنها تقسّم المنطقة L بالنسق SS . شكل هذه التعليقات هو التالي :

COP	L ₁	L ₂	B ₁	D ₁	B ₂	D ₂
-----	----------------	----------------	----------------	----------------	----------------	----------------

ونشير إلى أنه جرت العادة بالنسبة للتعليقات SS بأن تكون القيم المؤولة في المناطق L هي بالطول المذكور في تعليمة مؤول ناقص 1 . هكذا ، فالتعليمة :

A P ALPHA (16), BETA (10)

سيتم تأويلها مع القيم الثنائية 1111 و 1001 بالنسبة للطول .

تضع التعليقات الحسابية النتيجة في المتأثر الأول الذي يتم إلغاؤه ويجب أن يكون هذا المتأثر بطول كافٍ لاستيعاب النتيجة دون حدوث overflow وقطع للعدد . يظهر overflow إذا لم يكن المتأثر الأول بالطول المناسب لاستيعاب النتيجة . عندما تكون $L_1 < L_2$ لا يحدث overflow إذا لم يكن هناك مُرحّل (carry) خارج الإمكانات المقدمة من الطول L_1 . ويمكن تفنيع overflow بواسطة البتة SPM .

عند إجراء العمليات ، فإن الفاصلة لا تُمثّل والتراسف يتم لجهة اليمين ، كما يمكن حصر المتأثرات بواسطة عمليات إزاحة عشرية مناسبة .

تتحقق الدارات ، خلال التنفيذ ، من صلاحية الأرقام العشرية والإشارات .
 والتقاء عنصر غير صالح يؤدي إلى انقطاع من نوع استثناء بالمعطيات .
 المتأثرات 1 و 2 يمكن أن تندمج بشرط أن تكون بنفس المواقع (متراصفة) بالنسبة
 للبايتات ذات الوزن الأضعف . من الممكن هكذا إضافة عدد إلى نفسه :
 مثلاً :

ALPHA بعنوان

0	0	0	1	2	3	4	5	6	S
---	---	---	---	---	---	---	---	---	---

التعليمة :

S456 إلى S123456 AP ALPHA(5),ALPHA+3(2)

يتم تركيز كود الشوط CC حسب إشارة النتيجة .

2.16 . التعليقات

AP $D_1(L_1, B_1), D_2(L_2, B_2)$ SS COP=FA ADD DECIMAL
 $(S_1) + (S_2) \rightarrow (S_1)$
 يتم تركيز كود الشرط CC .

ZAP $D_1(L_1, B_1), D_2(L_2, B_2)$ SS COP=FB ZERO AND ADD
 $(S_2) \rightarrow (S_1)$
 تعادل العملية جمع عدد إلى صفر . ويتم تركيز CC .

SP $D_1(L_1, B_1), D_2(L_2, B_2)$ SS COP=FB SUBTRACT DECIMAL
 $(S_1) - (S_2) \rightarrow (S_1)$
 تركيز CC .

MP $D_1(L_1, B_1), D_2(L_2, B_2)$ SS COP=FC MULTIPLY DECIMAL
 $(S_1) \times (S_2) \rightarrow (S_1)$
 يجب أن نحصل على : $L_2 \leq 8$ و $L_2 < L_1$ وإلا سيحدث إنقطاع .
 CC يبقى بدون تعديل .

DP $D_1(L_1, B_1), D_2(L_2, B_2)$ SS COP=FD DIVIDE DECIMAL
 $(S_1) : (S_2) \rightarrow (S_1)$
 يجري وضع النتيجة إلى اليسار في (S_1) . الباقي يُخزّن إلى اليمين في (S_1)
 وينقص طول S_2 .
 حجم نتيجة القسمة هو 8 بتات : $L_2 - L_1$ يجب أن نحصل على $L_2 \leq 8$
 و $L_2 < L_1$ وإلا سيحدث إنقطاع ⁽¹⁾ CC بدون تعديل .

(1) إنتباه : يتعلق ذلك بالطول L بلغة المؤول وليس بطول القيم .

CP $D_1(L_1, B_1), D_2(L_2, B_2)$ SS COP=F9 COMPARE DECIMAL

تجري مقارنة المتأثرين ويتم تعديل مضمون CC . إذا كانت أطوال
'التأثرات غير متعادلة ، فإن المنطقة الأصغر يجري ملؤها بصفر لجهة
اليسار .

SRP $D_1(L_1, B_1), D_2(B_2), I_3$ SS COP=F0 SHIFT AND ROUND DECIMAL
(370)

يجب الانتباه إلى النسيق الخاص بهذه التعليمة . عند التأويل ، فإن I_3 تأخذ
الموقع الطبيعي المحفوظ لـ I_2 .

- S_1 هو عنوان المتأثر المطلوب إزاحته .

- L_1 هو الطول .

- $D_2(B_2)$ ليس عنواناً : البتات الست ذات الوزن الأضعف والمعتبرة

كعدد صحيح بإشارة ، تدل على اتجاه وعدد الأرقام العشرية المطلوب

إزاحتها . ويجري إهمال البتات الأخرى . القيمة السلبية (مكمل إلى

2) هي إزاحة إلى اليمين والنتيجة السلبية هي إزاحة إلى اليسار .

- I_3 هو « عامل التدوير » يُستعمل للإزاحات إلى اليمين . تضاف قيمته

إلى الرقم المستخرج بالإزاحة إلى اليمين والمُرَّحل المحتمل يرتد إلى

اليسار .

- توضع النتيجة في (S_1) .

- لا تشترك الإشارة بعملية الإزاحة .

17 الحساب بفاصلة متحركة

لم يبد لنا أساسياً شرح هذه التعليقات بكثير من العناية كما جرى بالنسبة للتعليقات السابقة . فدراسة هذه المجموعة من التعليقات لن تحمل لنا سوى قليلاً من المعلومات الجديدة حول الأولوية الأساسية لتشغيل المكثات ، بينما نحن نهتم بالدرجة الأولى بهذه الأولوية . ولكن المستعمل الذي فهم جيداً كل ما هو سابق لن ينزعج كثيراً من متابعة هذا الفصل . نفترض هنا بأن القارئ قد استوعب قراءة الفقرة 3.5.2 . ب حول الفاصلة المتحركة في تمثيل المعطيات . ولكي نتذكر بسهولة الكود الحرفي لهذه العمليات ، من الجيد أن نراجع الفقرة 2.10 المتعلقة بالترميز : الحرف النهائي «R» يختص بالتعليمة RR ، والأحرف E ، U ، D ، W و X هي نسق القصير المعايير (normalized) ، والقصير غير المعايير والطويل المعايير والطويل غير المعايير والموسّع .

1.17 . عموميات

هذه التعليقات تعمل مع المرافف المتحركة المرقمة 0 ، 2 ، 4 و 6 بطول 64 بته . الأعداد بفاصلة متحركة القصيرة توضع في الـ 32 بته ذات الوزن الأكبر من المرافف خلال العمليات . في هذه الحالة فإن الأوزان الضعيفة يجري إهمالها . الأعداد الطويلة بالفاصلة المتحركة تشغل كامل المرافف والأعداد الموسّعة بفاصلة متحركة تشغل مرصفين متتاليين . يجري تركيز موقع كود الشرط كالعادة :

جدول 1.17

CC	بالنسبة للتعليقات الجبرية	بالنسبة للمقارنات
0	نتيجة صفر	متأثر 1 = متأثر 2
1	نتيجة سلبية	متأثر 1 > متأثر 2
2	نتيجة إيجابية	متأثر 1 < متأثر 2
3		

2.17 التعليقات

يوجد نفس الخصائص التي رأيناها لدى معالجة الأعداد بفاصلة ثابتة . في حالة الشك بالإمكان مراجعتها

LER	R_1, R_2	RR	COP=38	LOAD	متأثرات قصيرة
LE	$R_1, D_2(X_2, B_2)$	RX	COP=78	LOAD	متأثرات قصيرة
LDR	R_1, R_2	RR	COP=28	LOAD	متأثرات طويلة
LD	$R_1, D_2(X_2, B_2)$	RX	COP=68	LOAD	متأثرات طويلة

CC دون تعديل

LTER	R_1, R_2	RR	COP=32	LOAD AND TEST	متأثرات قصيرة
LTDR	R_1, R_2	RR	COP=22	LOAD AND TEST	متأثرات طويلة
LCER	R_1, R_2	RR	COP=33	LOAD COMPLEMENT	متأثرات قصيرة
					شحن مع تغير الإشارة
LCDR	R_1, R_2	RR	COP=23	LOAD COMPLEMENT	متأثرات طويلة
					شحن مع تغير الإشارة
LNDR	R_1, R_2	RR	COP=31	LOAD NEGATIVE	متأثرات قصيرة
LNDR	R_1, R_2	RR	COP=21	LOAD NEGATIVE	متأثرات طويلة
LPER	R_1, R_2	RR	COP=30	LOAD POSITIVE	متأثرات قصيرة
LPDR	R_1, R_2	RR	COP=20	LOAD POSITIVE	متأثرات طويلة

تركيز أو تعديل CC

LRER	R_1, R_2 (370)	RR	COP=35	LOAD ROUNDED	التأثر 2 الطويل
					يجري تدويره ووضعه في التأثير الأول القصير
LRDR	R_1, R_2 (370)	RR	COP=25	LOAD ROUNDED	التأثر الموسع
					يجري تدويره ووضعه في التأثير الأول الطويل

دون تعديل

STE	$R_1, D_2(X_2, B_2)$	RX	COP=70	STORE	متأثرات قصيرة
STD	$R_1, D_2(X_2, B_2)$	RX	COP=60	STORE	متأثرات طويلة

CC دون تعديل

CER	R_1, R_2	RR	COP=39	COMPARE	متأثرات قصيرة
CE	$R_1, D_2(X_2, B_2)$	RX	COP=79	COMPARE	متأثرات قصيرة
CDR	R_1, R_2	RR	COP=29	COMPARE	متأثرات طويلة
CD	$R_1, D_2(X_2, B_2)$	RX	COP=69	COMPARE	متأثرات طويلة

تركيز أو تعديل CC

AER	R_1, R_2	RR	COP=3A	ADD NORMALIZED	متأثرات قصيرة
AE	$R_1, D_2(X_2, B_2)$	RX	COP=7A	ADD NORMALIZED	متأثرات قصيرة
ADR	R_1, R_2	RR	COP=2A	ADD NORMALIZED	متأثرات طويلة
AD	$R_1, D_2(X_2, B_2)$	RX	COP=6A	ADD NORMALIZED	متأثرات طويلة
AXR	R_1, R_2 (370)	RR	COP=36	ADD NORMALIZED	متأثرات موسعة

تركيز أو تعديل CC

AUR R_1, R_2	RR	COP=3E	ADD UNNORMALIZED (opq)	متأثرات قصيرة
AU $R_1, D_2(X_2, B_2)$	RX	COP=7E	ADD UNNORMALIZED (o	متأثرات قصيرة
AWR R_1, R_2	RR	COP=2E	ADD UNNORMALIZED (of	متأثرات طويلة
AW $R_1, D_2(X_2, B_2)$	RX	COP=6E	ADD UNNORMALIZED (op	متأثرات طويلة

تركيز أو تعديل CC

SER R_1, R_2	RR	COP=3B	SUBTRACT NORMALIZED	متأثرات قصيرة
SE $R_1, D_2(X_2, B_2)$	RX	COP=7B	SUBTRACT NORMALIZED	متأثرات قصيرة
SDR R_1, R_2	RR	COP=2B	SUBTRACT NORMALIZED	متأثرات طويلة
SD $R_1, D_2(X_2, B_2)$	RX	COP=6B	SUBTRACT NORMALIZED	متأثرات طويلة
SXR R_1, R_2 (370)	RR	COP=37	SUBTRACT NORMALIZED	متأثرات موسعة

تركيز أو تعديل CC

SUR R_1, R_2	RR	COP=3F	SUBTRACT UNNORMALIZED	متأثرات قصيرة
SU $R_1, D_2(X_2, B_2)$	RX	COP=7F	SUBTRACT UNNORMALIZED	متأثرات قصيرة
SWR R_1, R_2	RR	COP=2F	SUBTRACT UNNORMALIZED	متأثرات طويلة
SW $R_1, D_2(X_2, B_2)$	RX	COP=6F	SUBTRACT UNNORMALIZED	متأثرات طويلة

تركيز أو تعديل CC

MER R_1, R_2	RR	COP=3C	MULTIPLY	متأثرات قصيرة ونتيجة موسعة
ME $R_1, D_2(X_2, B_2)$	RX	COP=7C	MULTIPLY	متأثرات قصيرة ونتيجة موسعة
MDR R_1, R_2	RR	COP=2C	MULTIPLY	متأثرات طويلة
MD $R_1, D_2(X_2, B_2)$	RX	COP=6C	MULTIPLY	متأثرات طويلة
MXDR R_1, R_2 (370)	RR	COP=27	MULTIPLY	متأثرات طويلة ونتيجة موسعة
MXD $R_1, D_2(X_2, B_2)$ (370)	RX	COP=67	MULTIPLY	متأثرات طويلة ونتيجة موسعة
MXR R_1, R_2 (370)	RR	COP=26	MULTIPLY	متأثرات موسعة

دون تعديل

DER R_1, R_2	RR	COP=3D	DIVIDE	متأثرات قصيرة
DE $R_1, D_2(X_2, B_2)$	RX	COP=7D	DIVIDE	متأثرات قصيرة
DDR R_1, R_2	RR	COP=2D	DIVIDE	متأثرات طويلة
DD $R_1, D_2(X_2, B_2)$	RX	COP=6D	DIVIDE	متأثرات طويلة

لا تتغير CC

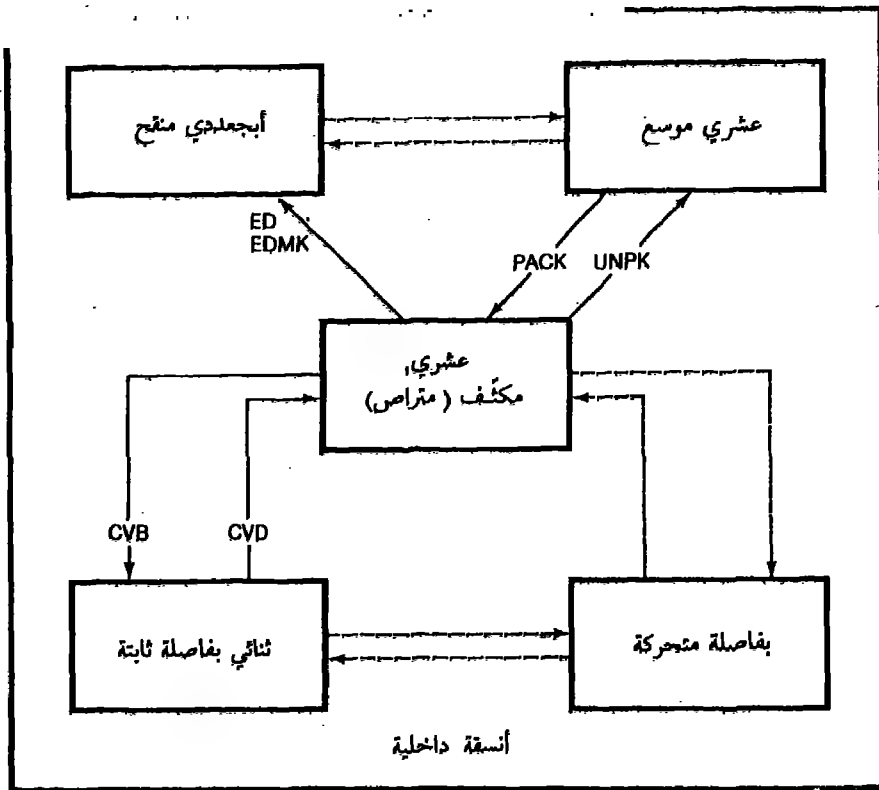
HER R_1, R_2	RR	COP=34	HALVE	متأثرات قصيرة
HDR R_1, R_2	RR	COP=24	HALVE	متأثرات طويلة

يُقسم المتأثر الثاني على 2 وتوضع نتيجة القسمة المعبرة في المتأثر الأول .

18 . تعليمات التحويل والتمثيل

1.18 . عموميات

لقد رأينا أن النظام 370 كان يتمتع بثلاث طبقات من الدارات الحسابية العاملة بثلاث طرق مختلفة لتمثيل المعطيات الرقمية . ولكن ، المعطيات الداخلة إلى الذاكرة تكون عادةً مكوّنة بتمثيل أبجدي . من هنا ، فإن كل عملية حسابية على معطى رقمي داخل إلى المكنة ، من خلال ناقل بطاقات مثلاً ، يمكن أن تتطلب عدة عمليات تحويل للتمثيل قبل معالجته بالحساب العشري ، الثنائي أو بفاصلة متحركة . المخطط 1.18 يعرض مختلف الأشكال الداخلية وعمليات النقل الممكنة التي تتم بواسطة هذه التعليمات . الخطوط المنقطة تمثل التحويلات التي تجريها برامج متخصصة .



خطوط 1.18

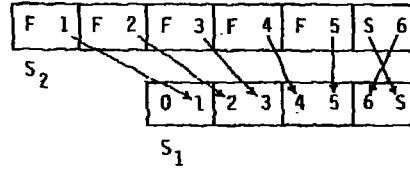
2.18 . تعليمات التحويل

PACK $D_1(L_1, B_1), D_2(L_2, B_2)$ SS COP=F2 PACK
(S_2) \rightarrow (S_1)

عشري مكثف عشري موسع
(متراس)

هذه التعليمة تحول منطقة S_2 ، يفترض إنها عشرية موسعة ، إلى عشرية متراسة . التحويل يتم من اليمين إلى اليسار بدون تحقق من صلاحية الأكواد .

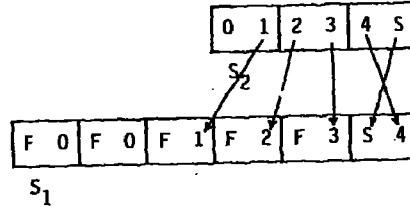
إذا كانت المنطقة S_1 أكبر من الضروري ، فهي تكمل بأصفار (00) لجهة اليسار .
إذا كانت S_1 قصيرة جداً يحدث قطع لجهة اليسار .
 S_1 و S_2 يمكن أن تتراكبا .



UNPK $D_1(L_1, B_1), D_2(L_2, B_2)$ SS COP=F3 UNPACK
(S_2) \rightarrow (S_1)

عشري موسع عشري مكثف

التعليمة تحول منطقة S_2 ، يفترض إنها عشرية متراسة ، في S_1 عشري موسع
التحويل يتم من اليمين إلى اليسار ، بدون تحقق من صلاحية الأكواد .
إذا كانت المنطقة S_1 أصغر ، يحدث قطع أو بتر لجهة اليسار .
إذا كانت طويلة تستكمل بأصفار (F0) لجهة اليسار .
 S_1 و S_2 يمكن أن تتراكبا .



CVB $R_1, D_2(X_2, B_2)$

RX COP=4F CONVERT TO BINARY
(S_2) \rightarrow R_1

ثنائي عشري متراس
محصورة في كلمة مزدوجة

صلاحية الإشارة والبنات الرقمية في S2 يتم التحقق منها . كل خطأ يؤدي إلى انقطاع .

يفترض بأن تكون S2 عبارة عن عنوان لكلمة مزدوجة بطول 8 بايتات . يُجَدُّ التحويل بالأعداد القصوى والصُغرى التي من الممكن تمثيلها في 32 بته ، أي :

$$-2\ 147\ 483\ 648 \leq +2\ 147\ 483\ 647.$$

CVD R₁,D₂(X₂,B₂) RX COP=4E CONVERT TO DECIMAL
R₁ → (S₂)

عشري متراص
موجود في كلمة مزدوجة

ثنائي

يتألف العدد العشري الحاصل من 15 رقماً إضافة إلى الإشارة : «C» للجمع (+) و«D» للنقص (-) . يبقى كود الشرط بدون تغيير .

3.18 . التنقيح والطباعة

إنّ مضمون كلمة آلية ثنائية ، لمُعْطى عشري أو بفاصلة متحركة يجب ، قبل طباعته أن يخضع لتحويل معيّن . يجب أن يتم تحويل قيمته الثنائية إلى أكواد من السّيات القابلة للطباعة . قد يكون من الضروري إدخال فاصلة ، نقطة عشرية ، إشارة أو سيات تعبئة (حالة طباعة الشيكات) .

يوجد تعليمتان ED وEDMK تحقّقان هذا العمل بتحويل منطقة أولية (عشري متراص) إلى منطقة تنقيح وطباعة .

مثلاً :

منطقة أولية

0	0	1	2	3	4	5	D
---	---	---	---	---	---	---	---

منطقة تنقيح

5	C	5	C	5	C	6	0	F	1	F	2	F	3	4	B	F	4	F	5
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

* * * - 1 2 3 . 4 5

سيات تعبئة إدخال فاصلة عشرية

لكي يتم هذا ، فإن المبرمج يضع في جيّز الطباعة قناعاً مؤلفاً من :

- سمة تعبئة .

- أكواد تدل على : مواقع الأرقام ، المكان الذي من خلاله يتم تحويل الأصفار «0» بدون ذات معنى ، السيات المطلوب إدخالها في نهاية حقل الطباعة .

هذه التعليمات تعمل بعلاقة مع مؤشر ثنائي يُدعى « مؤشر معني » . يُوضع هذا المؤشر في «1» عندما نلتقي برقم ذي معنى في المنطقة الأولية أو عندما نلتقي مكان الأصفار التي من الواجب تحويلها .

تعرّف هنا على العمل الجاري بواسطة « صور » الطباعة بلغة كويول . لن يتم شرح هذه التعليقات هنا ونصبح بمراجعة وثائق IBM370 .

ED $D_1(L, B_1), D_2(B_2)$ SS COP=DE EDIT

S_1 : منطقة الطباعة ، بطول L وتحتوي حل القناع .
 S_2 : عنوان المنطقة الأولية (المنبع هو منطقة عشرية مشراصة) . يتم تعديل CC حسب إشارة آخر حقل .

EDMK $D_1(L, B_1), D_2(B_2)$ SS COP=DF EDIT AND MARK

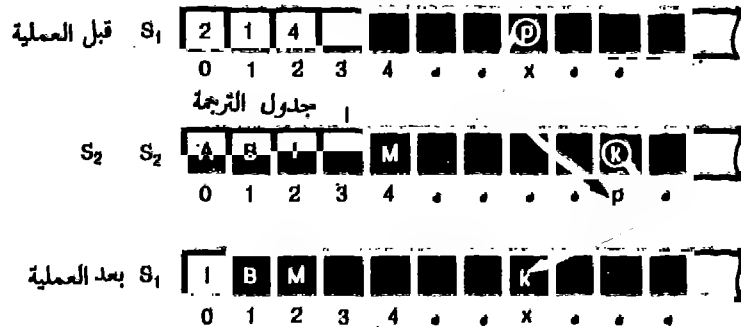
تتمتع S_1 و S_2 بنفس المعنى . عنوان الرقم الأول ذي المعنى يُخزّن في المرصف 1 .
 يتم تعديل مضمون CC حسب إشارة آخر حقل .

4.18 . الترجمة

TR $D_1(L, B_1), D_2(B_2)$ SS COP=DC TRANSLATE

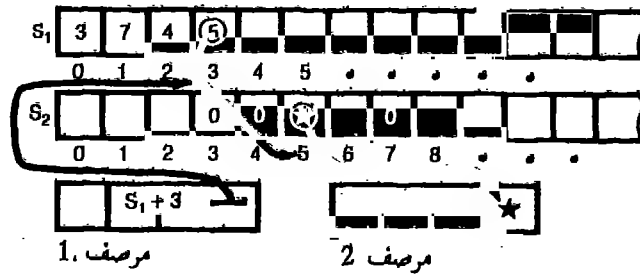
ترجمة سلسلة (S_1) بطول L حسب جدول موجود في S_2 بطول أقصى يبلغ 256 بايتة .

قبل العملية ، فإن البايته ($0 \leq x < L$) $S_1 + x$ تحتوي على الرقم p ($0 \leq p < 255$) الذي يستخدم كنقطة إدخال إلى الجدول .
 بعد العملية : $(S_2 + p) \rightarrow (S_1 + x)$ يبقى CC بدون تعديل .



S_1 : منطقة البحث بطول L .
 S_2 : عنوان جدول الترجمة .
 التعليمة تستعمل المرصفين 1 و 2 .
 تؤخذ البايته الأولى من المنطقة S_1 بعين الاعتبار . كما في TR ، فإن قيمته الشائبة تشكل نقطة دخول في S_2 .

إذا كانت البايته المناسبة S_2 مختلفة عن صفر فإن قيمتها تُخزَّن في المرصف 2 وعنوان المنطقة التي تسمح بإيجاد التناسب يُخزَّن في المرصف 1 .
 وإلا فإن العملية تتابع مع البايته التالية من S_1 . يتم تركيز $CC = 0$ إذا كانت المنطقة S_1 قد جرى إستكشافها كلياً وبجميع البايتات التي جرى إختيارها من S_2 كانت صفراً .
 $CC = 1$ إذا جرى إستكشاف S_1 بشكل جزئي ولم تكن البايته الأخيرة المختارة صفراً .
 $CC = 2$ إذا جرى إستكشاف المنطقة كلياً وكانت البايته الأخيرة المختارة مختلفة عن صفر .



$CC = 1$
 في هذا المثل ، لنستطيع متابعة إستكشاف المنطقة ، يجب إعتداد تعديل
 لعنوان الانطلاق والطول المستكشف .
 $R_1(0-7)$ و $R_2(0-23)$ يبقيان دون تعديل .
 S_1 لا يتم تعديلها .

تمارين :

تمرين 1.18 - إعادة تنظيم منطقة من الذاكرة .

لنفترض منطقة ARTICLE من 10 بايتات نرغب بنقل البايتات 5 ، 6 ، 7 ،
1 ، 2 إلى المنطقة CLE

ARTICLE	A	B	C	D	E	F	G	H	I	J
CLE	F	G	H	B	C					

اكتب التعليمات التي تسمح بإجراء هذا العمل . في نفس الفكرة نرغب
بعكس سلسلة من النسات . هذا النظام يستعمل لاعادة تنظيم مفاتيح
الفرز .

تمرين 2.18 - لنفترض منطقة مؤلفة من 8 بايتات بقيم ثنائية موجودة بين 0 و15 . نرغب
باستبدالها بالكود EBCDIC المناسب للقيم السادس عشرية : سيجري
إستبدال 0 بواسطة '0C' ، و10 بواسطة 'A'C' ... اكتب التعليمات
المناسبة .

هذه الأوامر يمكن أن تستعمل ، بعد عملية تحويل بسيطة ، لطباعة
مضمون سادس عشري لكلمة من الذاكرة ، للتحضير للطباعة بواسطة
DUMP (دلق) .

19 الانقطاع والادخال والاخراج

(Interruptions and I/O)

1.1.19 . الانقطاعات

لن يكون موضوعنا تفصيل نظام إدارة الانقطاعات هنا ، ولكن فقط إعطاء القارئ إشارات بالنسبة لطبيعة هذه المسألة . لتفصيلات أكثر تنصح بمراجعة وثائق المنشئ Principles of operation .

1.1.19 . صيغة الانقطاعات

الانقطاع هو عبارة عن إشارة كهربائية ، مُرسلة من أحد أعضاء النظام ومعروفة من قبل الوحدة المركزية . ينتج الانقطاع عن حادثة تتطلب عادةً معالجة مباشرة . لبعض الحوادث صفة خاصة مستعجلة تتطلب تعليق دوران تنفيذ أحد البرامج الجارية كي يتم معالجة الإشارة المُرسلة . في النظام IBM 370 ، الحوادث القادرة على تفريع ووقف تنفيذ البرنامج قد جرى تصنيفها حسب أولوية متناقصة :

- نداء للمشرف (call supervisor) ،

- برنامج ،

- عطل في المكنة ،

- إشارة خارجية ،

- عملية إدخال - إخراج (I/O) ،

- إشارة مؤثر (operator signal) .

يرتبط بكل فئة درجة إستعجال معينة . نتكلم هنا عن ستة مستويات من الانقطاعات ونظام معالجة الحوادث يجري حسب الأولوية المعتمدة .

2.1.19 . أولية الإنقطاع

نذكر بأن المفهوم الذي يدور حوله البرنامج مؤلف من كلمة حالة البرنامج PSW ومن مضمون المرافف العامة والمتحركة المرتبطة به . نشير أيضاً إلى أنه في كل لحظة ، PSW تحتوي على القيمة الحالية لعداد البرنامج . يؤدي تعليق دوران البرنامج أوتوماتيكياً

إلى تخزين مضمون هذه المرافف كي نستطيع معاودة تنفيذ هذا البرنامج المقطوع عند الحاجة . هكذا فالانقطاع يؤدي إلى إطلاق العملية التالية :

1- بشكل أوتوماتيكي (أي بواسطة العتاد (hardware)) ، فإن وصول إشارة الانقطاع تؤدي إلى نسخ PSW الخاصة بالبرنامج الجاري في منطقة محدّدة من الذاكرة ، تُمَيِّز فئة الانقطاع . تدعى هذه الكلمة PSW « الكلمة القديمة » .

2- بشكل أوتوماتيكي ، يأخذ العتاد على عاتقه الكلمة الجديدة PSW الموجودة على عنوان من الذاكرة حسب فئة الانقطاع . منذ هذه اللحظة ، يمكن تنفيذ برنامج جديد : وتبدأ معالجة الانقطاع .

3- بعد الإنتهاء من معالجة الانقطاع ، يمكن معاودة العمل بالبرنامج المقطوع وذلك بواسطة إعادة ترميم الكلمة PSW وإعادة تخزين المرافف بالمعلومات التي كان يحتويها قبل قطع البرنامج .

نضيف أن معالجة الانقطاع يمكن أن تُقطع بدورها بواسطة حادثة أكبر أولوية . مجموعة البرامج التي تعالج الانقطاعات تعتبر جزءاً من نظام التشغيل وتدعى نظام إدارة الانقطاعات .

3.1.19 . قناع الانقطاعات

هذه الأولوية الأساسية يمكن ، ضمن بعض الشروط ، أن يتم « تقنيها » بواسطة المبرمج . بواسطة تفسير الأتعة في الكلمة PSW يمكن للمبرمج أن يمنع أخذ الحوادث الطارئة بالحساب . هكذا يمكن إهمال الفيض overflow الناتج عن الحساب وذلك بتركيز القناع المناسب بواسطة التعليمات SPM . الإنقطاع المبرمج المُقنّع لا يتم أبداً ، كما يوضع الإنقطاع المُقنّع الناتج عن النظام في الانتظار حتى يجري رفع القناع أو القيد عنه . التعليمات SSM التي تسمح بتعديل قناع النظام هي تعليمات خاصة .

4.1.19 . الانقطاعات الناتجة عن البرنامج

سنعطي هنا أسباب الانقطاعات الناتجة عن البرنامج . وهي تولّد عادة بسبب خطأ في البرمجة . وتجري الإشارة إليها بواسطة ظهور كود للعودة OCx يُدعى «completion code» أو كود الانتهاء .

لتفاصيل أكثر يجب على القارئ أن يراجع وثائق IBM الخاصة .

OPERATION EXCEPTION

code = 0C1

يتج هذا الانقطاع عندما يكون هناك محاولة لتنفيذ تعليمات بكود عملية غير صالح .

PRIVILEGED-OPERATION EXCEPTION

code = 0C2

محاولة لتنفيذ تعليمة خاصة بينما تكون المكنة في صيغة المسألة .

EXECUTE EXCEPTION

code = 0C3

التعليمة EX تعود إلى تعليمة أخرى EX .

PROTECTION EXCEPTION

code = 0C4

يتعلق ذلك ببلوغ موقع محمي من الذاكرة .

ADRESSING EXCEPTION

code = 0C5

يتعلق ذلك بمحاولة بلوغ موقع غير موجود في الذاكرة .

SPECIFICATION EXCEPTION

code = 0C6

هذا الانقطاع يغطي أكثر الحالات ، لن نذكر سوى الأكثر شيوعاً . يتعلق ذلك بمسألة الحدود : لا تحصر التعليمة بحدود نصف كلمة أو معطى غير مسطر كما تحتاج التعليمة التي تُرجع إليها .

DATA EXCEPTION

code = 0C7

يتعلق ذلك بمشكلة ناتجة عن تعليمة CVB أو تعليمة عشرية .

FIXED-POINT-OVERFLOW EXCEPTION

code = 0C8

overflow في تمثيل بفاصلة ثابتة .

FIXED POINT DIVIDE EXCEPTION

code = 0C9

يتعلق ذلك بالقسمة على صفر ، أو بنتيجة قسمة يزيد حجمها عن حجم المرصف أو بتحويل إلى ثنائي (CVB) حيث النتيجة تزيد عن 31 بتة .

DECIMAL-OVERFLOW EXCEPTION

code = 0CA

نلتقي هذه التعليمة في عملية على أعداد عشرية ، عندما يتم فقدان البتات ذات الأوزان العليا لأن المنطقة النهائية هي أصغر من أن تحتوي على النتيجة .

DECIMAL-DIVIDE EXCEPTION

code = 0CB

يتعلق ذلك بالقسمة على صفر في عملية بالنظام العشري .

EXPONENT-OVERFLOW EXCEPTION

code = 0CC

الأس الخاص بالنتيجة يزيد عن 127 والقسم العشري (mantisse) ليس صفراً .

EXPONENT-UNDERFLOW EXCEPTION

code = 0CD

الأس هو سلمي والقسم العشري ليس صفراً .

SIGIFICANCE EXCEPTION

code = 0CE

في عملية جمع أو طرح على أعداد بفاصلة متحركة والقسم العشري هو صفر .

FLOATING POINT-DIVIDE EXCEPTION

code = 0CF

قسمة على صفر لأعداد بفاصلة متحركة .

5.1.19 . تعليمات مرتبطة بالانقطاعات

SPM R₁

RR COP=04

SET PROGRAM MASK

R₁(2-7) → CC,

أقنعة البرنامج



0 8 15

البتات من 2 إلى 7 من المرصف العام R₁ تُخزّن (البتات 2 و3) في CC وفي (البتات 4 و7) قناع البرنامج . نشير هنا إلى أن التعليمات BAL وBALR تشحن المرصف R₁(2-7) بالكود CC ويقناع البرنامج .

SVC

RR COP=0A

SUPERVISOR CALL

هذه التعليمة تؤدي إلى انقطاع بكود I . الكلمة القديمة PSW تُخزّن في الذاكرة على العنوان 32 والكلمة الجديدة PSW تؤخذ على العنوان 96 .

MC
(370)

D₁(B₁), I₂

SI COP=AF

MONITOR CALL

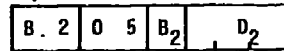
تطلق برنامج انقطاع عندما تكون بته خاصّة من القناع الموجه في 1 .

STCK
(370)

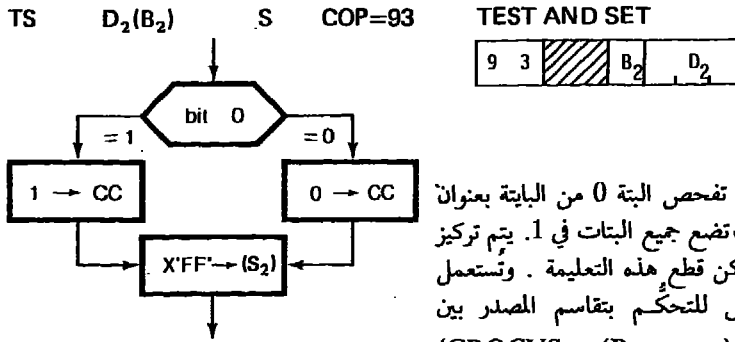
D₂(B₂)

S

COP=B205 STORE CLOCK



مة الحالية للساعة توضع في كلمة مزدوجة بعنوان S₂ . البتة 31 من ساعة تزداد كل 1,048566 ثانية . ويتم تركيز كود الشرط حسب حالة الساعة .



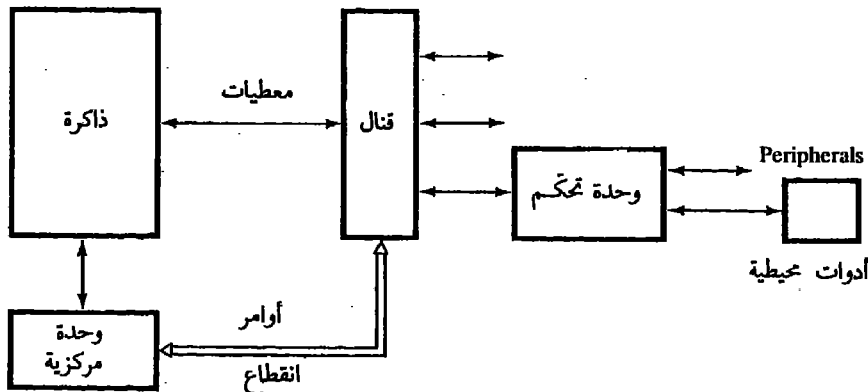
هذه التعليمة تفحص البتة 0 من البايته بعنوان S₂ وبعد ذلك تضع جميع البتات في 1. يتم تركيز CC . لا يمكن قطع هذه التعليمة . وتستعمل بشكل خاص للتحكم بتقاسم المصدر بين عمليتين (Processus) و (CROCUS) systemes des exploitation des ordinateurs, Dunod)

2.19 . الإدخال - الإخراج

سنعرض هنا للعمليات المهمة لإجراء المداخل والمخارج . بإمكان القارئ ، عند القيام باختباراته ، إجراء إدخال - إخراج باستعمال حلقات من فورتران ، مثلاً ، أو بفضل وجود ماكرو تعليقات موجودة على النظام الذي يعمل عليه . سنعود بعد قراءة العموميات إلى دراسة ماكرو تعليقات الإدخال - الإخراج .

1.2.19 . تعريف وأولية الإدخال - الإخراج

عملية الإدخال - الإخراج هي عملية نقل المعطيات من الذاكرة إلى الأدوات المحيطة وبالعكس وتتم بأمر من الوحدة المركزية تحت مراقبة وتنفيذ القنال .



عند إطلاق العملية فإنها تدور دون تدخل الوحدة المركزية . يظهر القنال وكأنه معالج مستقل ويخصص لتبادل المعطيات بين الذاكرة والجهاز المحيطي . وبشكل عام ،

يوضع البرنامج الذي طلب الإدخال / الإخراج في الانتظار حتى إنتهاء عملية الإدخال / الإخراج . وهذا يعني أن تنفيذه معلق خلال مدة الإدخال / الإخراج . وهو يفقد مصادر الوحدة المركزية التي يمكن أن تُخصَّص إلى برامج أخرى مُنتظرة التنفيذ . بعد إنتهاء عملية الإدخال - الإخراج - وهذا ما يتم إعلام النظام به بواسطة الإنقطاع - سيكون بإمكان البرنامج المقطوع أن يُعاود العمل ، وسيوضع في سجل البرامج التي تنتظر مصادر الوحدة المركزية . هنا يدخل موضوع المزامنة المفروض من الإدخال - الإخراج . يتم تأمين هذا التنظيم والإدارة بواسطة برامج (رُجل) خاصة من نظام التشغيل وهذا هو السبب الذي لأجله لا يستطيع المبرمج أن يُوجِّه بالكامل عمليات الإدخال - الإخراج الخاصة به . فهو يعطي فقط الإشارات اللازمة لنظام التشغيل ليؤمن حسن تشغيل ودوران برنامجه .

2.2.19 . المعلومات الضرورية لعملية إدخال - إخراج
فلنفكر من خلال مثل من فورتران . لنفترض عملية كتابة على الطابعة I و J هي متحولات صحيحة .

```
WRITE(6,1000) I,J
1000  FORMAT(1X,'I= ',I5,'J= ',I5)
```

إذا كانت قيمة I و J هي على التوالي -4532 و 3 ، نحصل إذاً على :

$I = \Delta -4532 \Delta J = \Delta\Delta\Delta\Delta\Delta 3$

حيث Δ ترمز إلى الفسحة (البياض) الفارغة .

هذه التعليمة في الإدخال - الإخراج المستوحاة من لغة متطورة تغطي مرحلتين مختلفتين .

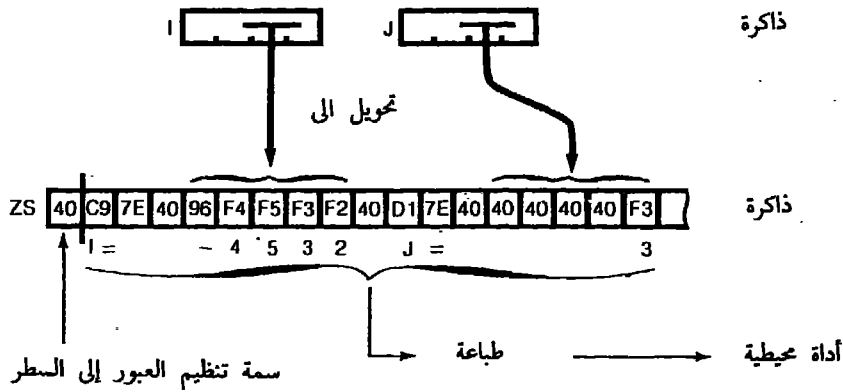
- لتحويل المتحولات الصحيحة I و J (ثنائي بفاصلة ثابتة) إلى سمات قابلة للطباعة .

وؤمن عملية الإدخال - الإخراج ، أي تبادل المعطيات .
المخطط اللاحق يُوجز العمليات .

النسق FORMAT يُثَّل إذاً القناع الذي تكلمنا عنه عند دراسة تعليقات الطابعة . المرحلة 1 تتم تحت تحكم البرنامج ، المرحلة 2 تقع على عاتق القنال .

نلاحظ إذاً أنه من الضروري معرفة :

- نوع الأداة المحيطية (رقم الوحدة المنطقية ، بلغة فورتران) ،
- العنوان ZS للمنطقة المطلوب طباعتها .



- طول ZS بالبايتات ،
- نوع الأمر (READ أو WRITE) .

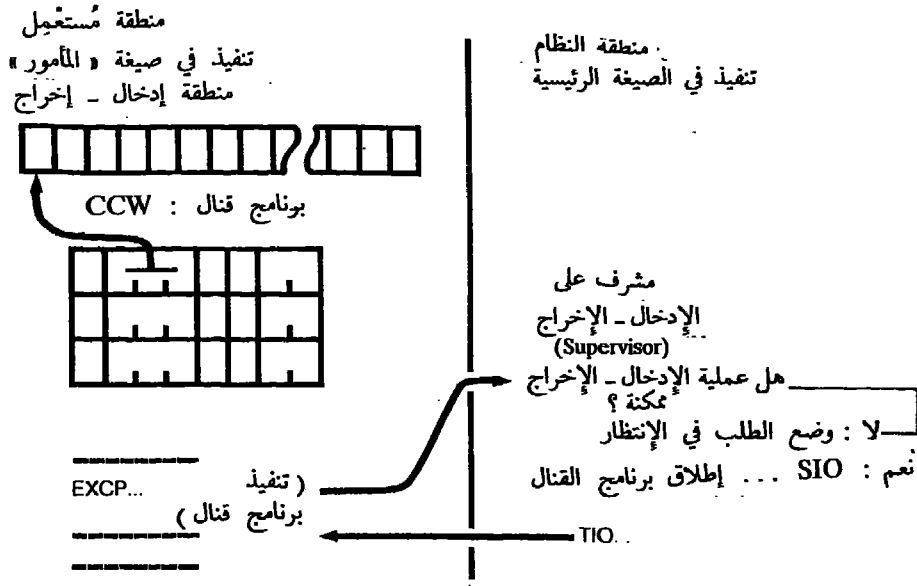
هذه المعلومات إضافة إلى معلومات أخرى ، لأن عمليات الإدخال - الإخراج هي في الواقع أكثر تعقيداً ، يتم وضعها في كلمة مزدوجة للتحكم بالقناة تدعى CCW (Channel command word : كلمة أمر للقناة) .

يلعب القنال دور الحاسب لأنه قابل للبرمجة . ستدعى « برنامج قنال » أو « برنامج وحدة تبادل » ، مجموعة الكلمات CCW المكونة من أوامر متتالية تتحكم بالمحيط .

الأدوات المحيطية هي عبارة عن مصادر قابلة للتقاسم والتوزيع بين عدة مستعملين . يصبح إذاً من الضروري معالجة النزاعات التي قد تولد من جراء طلبات متزامنة لنفس المصدر . لهذا السبب فإن مسؤولية إطلاق برنامج القنال تقع على عاتق نظام التشغيل الذي سيتحقق من توفر القنال والوحدة المحيطية . وبشكل آخر ، بإمكانه أن يأخذ بعض القرارات في حالة حدوث تنفيذ خاطئ لعملية الإدخال - الإخراج . الكلمة - المزدوجة ذات العنوان 40 ، بالنظام السادس عشري ، والتي تدعى (Channel status word) CSW ، تعطي بعض المعلومات حول دوران ومحاولة إطلاق الإدخال - الإخراج . المخطط الوارد على الصفحة التالية يقوم ببعض عمليات الربط بين مختلف العناصر الضرورية للإدخال - الإخراج .

3.2.19 . إدخال - إخراج في المستوى المنطقي

إن تنفيذ عملية إدخال - إخراج بالمستوى الفيزيائي هو أمر معقد . كتابة CCW تتطلب معرفة واضحة بالمحيطات التي نعمل عليها . ونعرف أنه في أغلب الوقت تكون



عمليات الإدخال - الإخراج على المحيطات البسيطة مؤجلة . عندما يقوم المستعمل بتعريف سجل طباعة (حالة (6,...) WRITE بلغة فورتران) ، فإن هذا السجل هو أولاً مكتوب على قرص مغناطيسي وبعد ذلك ، بواسطة برنامج خاص ، يُؤخذ لإجراء طباعة نهائية . وفي المجموع فإن رقم الوحدة المنطقي ، يُناسب أولاً فيزيائياً سجل قرص مغناطيسي وبعد ذلك سجل طباعة . هذه العملية ، التي تحاول تبسيط إدارة المصادر المركزية والمحيطية ، تؤدي إلى زيادة الصعوبة في تنفيذ عملية الإدخال - الإخراج الفيزيائية . من جهة أخرى ، فإن تنظيم عملية إدخال - إخراج يؤدي إلى درء (Bufferization) لمناطق إدخال - إخراج . نعرف أيضاً أنه يوجد عدة تنظيمات نموذجية للسجلات وعدة طرق للبلوغ . هذه الشروط تفرض على المستعمل بأن يأمن بالكامل لنظام إدارة عمليات الإدخال - الإخراج . للقيام بذلك يجب عليه وصف المتغيرات الوسيطة المفيدة بواسطة توجيه من نوع DCB (DATA CONTROL BLOCK) . وهو سيوكل عملية الإدخال - الإخراج الخاصة به للنظام بواسطة ماكرو تعليمة خاصة (PUT GET, ...) حسب نوع تنظيم السجل الخاص به . هذه الأخيرة هي موضحة في الوثائق OS/VS2 MVS (DATA Management Macro Instructions) . يقوم النظام بتوليد الكلمات CCW لنفسه ونداء المشرف الضروري . العملية الأولى للإدخال - الإخراج ستكون مسبقة بفتح للسجل (ماكرو OPEN) والأخيرة ستكون متبوعة بإغلاق للسجل (ماكرو CLOSE) يسمح بتفريغ الداريء (Buffer) الأخير . المثل التالي يوضح ، بإشراف النظام OS ، عملية قراءة بطاقة مثقوبة وكتابة على الطباعة .

```

OPEN      (CARTE,(INPUT))
OPEN      (IMP,(OUTPUT))
-----
GET       CARTE,ZENTREE
-----
PUT       IMP,ZSORTIE
-----
CLOSE     CARTE
CLOSE     IMP
-----
CARTE     DCB      DDNAME=ENTREE,DSORG=PS,LRECL=80,BLKSIZE=400,MACRF=(GM),
                   RECFM=FB,EODAD=SUIE
IMP       DCB      DDNAME=SORTIE,DSORG=PS,LRECL=133,BLKSIZE=665,MACRF=(PM),
                   RECFM=FBA
ZENTREE   DS       CL'80'
ZSORTIE   DC       133C' '
-----

```

20 . الأوامر المتعلقة بالعنونة

وتركيبة البرنامج

سنقوم بجمع الأوامر (التوجيهات) المستعملة عند بداية ونهاية البرنامج ، التي تسمح بإعداد عداد المواقع ، وتعريف المراسف القاعدية أو تغيير وتقطيع البرامج .

1.20 . تعريف وشحن مراسف القاعدة

لقد عرفنا العنونة القاعدية (فقرة 2.3) وعرضنا مثلاً على تأويل تعليمة من هذا النوع (فقرة 3.3.6) من الضروري العودة الآن بشكل أكثر تنظيمياً لهذه المسألة :

إهتمامات المبرمج الأولى هي :

- 1- تحديد واحد أو عدة عناوين قاعدية .
- 2- حجز واحد أو عدة مراسف سيتم استعمالها كمراسف قاعدية .
- 3- شحن هذه المراسف بالعناوين المناسبة .

النقطتان الأوليان تتعلقان بمرحلة التأويل ، والنقطة الثالثة تتعلق بمرحلة التنفيذ ولا يمكن أن نُحلَّ بشكل نهائي عند التأويل لأن العنوان الفعلي لحزن البرنامج في الذاكرة لن يكون معروفاً إلا في لحظة الشحن .

أ - USING

هو الأمر الذي يسمح للمؤول بتحديد مراسف القاعدة وحساب الإزاحة المطلوبة لعنوان محدّد رمزياً (قاعدة ضمنية ، فقرة 2.9) . وشكله هو التالي :

USING Ad. base; numero des registres de base

رقم مرصف القاعدة وعنوان قاعدة USING

«Ad. Base» هو تعبير مطلق أو قابل للنقل يعتبره المؤول عنواناً قاعدياً . هذا الأمر لا يُؤلّد أية تعليمة ولذلك فهو لا يزيد من قيمة عداد المواقع . وهو يختفي من البرنامج المؤول .

مثلاً :

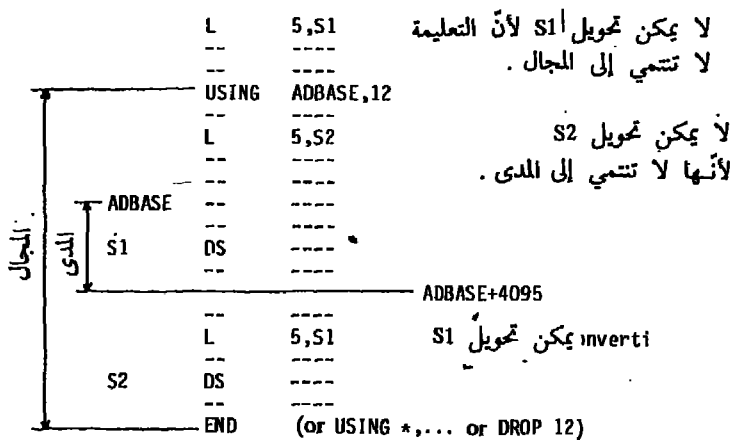
- (1) USING ADBASE,12
- (2) USING ADBASE,12,11,10
- (3) USING *,15

الإزاحة هي كمية مكوّنة من 12 بة لا تزيد عن 4095 . وبالتالي ، فإن مدى مصرف القاعدة 12 سيمتد من ADBASE إلى ADBASE + 4095 . عندما يزيد البرنامج عن 4096 بايتة يجب إستعمال الشكل (2) أو عدة أوامر USING لتحقيق العنونة . في الشكل (2) يفترض المؤول أنّ المصرف 12 يحتوي على القيمة ADBASE ، والمصرف 11 القيمة ADBASE+4096 والمصرف 10 القيمة ADBASE + 8192 . في الشكل (3) يفترض المؤول إن العنوان القاعدي هو القيمة الحالية لعدد المواقع .

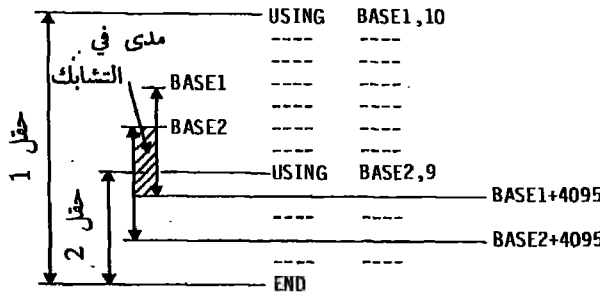
قواعد الإستعمال

لنميز « مدى » المصرف القاعدي من الحقل المغطى بواسطة تعليمة USING . مدى المصرف القاعدي لا يتعلّق سوى بالعنوان القاعدي المذكور في الأمر وليس بموقع USING . ويمتد من ADBASE إلى ADBASE+4095 . هذا يعني إن جميع الرموز التي تنتمي إلى المنطقة يمكن أن تعنّون بناء على انتهاء التعليقات التي ترجع إليها إلى « الحقل » .

الحقل USING يمتد من الأمر (التوجيه) USING حتى نهاية (END) الزجلة . الأمر الآخر USING يُحدّد نفس المصرف أو يضع الأمر DROP النهاية للحقل السابق . المثل التالي يوضح ذلك .



حالة استعمال عدة أوامر USING عندما يتشابه مدى عدة مرصيف ، فإن المؤول يحدد بشكل جلي العناوين الرمزية المشتركة لكلا المدينين باختيار عنوان قاعدي ذلك الذي ينتج أصغر إزاحة . إذا كانت العناوين القاعدية متشابهة (BASE1 و BASE2 هي ذاتها) ، فهو يختار رقم المرصيف الأكبر . إذا كانت العناوين مختلفة ولكن المرصيف متشابهة فإن الأمر الثاني USING يقطع مدى الأول



ب - شحن مرصيف القاعدة

يتوجه الأمر USING إلى مرحلة التأويل (assembling) . يجب على المبرمج أن يتوقع تعليمة تقوم ، عند التنفيذ ، بتخزين المرصيف القاعدية بالعناوين الفعلية الضرورية . هذه العناوين لا يمكن أن تكون معروفة في لحظة التأويل (assembling) لأنها تتعلق بنقطة الشحن (فقرة 4.6) . المشكلة هي إذاً في كيفية معرفة طريقة استرجاع هذه العناوين . نستعمل لذلك تقنيتين : الطريقة الأولى تستعمل حالة خاصة في استعمال BALR : حيث R2 هو المرصيف 0 (فقرة 4.12) . هكذا فمن الممكن كتابة :

```
BALR 12,0
USING *,12
```

يُخزّن عنوان التعليمة BALR زائد 2 (طول التعليمة) في المرصيف 12 وهذا العنوان (*) يُحدّد كقاعدة .

الطريقة الأخرى تقوم على إستعمال إتفاق عادي من النظام OS (فقرة 5.21) بوجهه يُخزّن النظام في المرصيف 15 عنوان نقطة الدخول إلى البرنامج الذي ينتقل التحكم إليه . هذه هي طريقتنا المفضلة . سنختار كعنوان قاعدي عنوان بداية (نقطة الدخول) إلى البرنامج .

DEBUT CSECT

USING DEBUT,12
LR 12,15

وبالتالي ، وحدها التعليقات التي لا تستعمل عناوين رمزية يمكن أن تظهر قبل شحن المرصف القاعدي .

ج - DROP

التوجيه أو الأمر DROP R₁, R₂, ... R_n يُشير إلى المؤول لكي لا يستعمل المراصف R₁, R₂, ... R_n كمراصف قاعدية .

2.20 . تقطيع البرامج

كل برنامج مهم يجب أن يكون مقطوعاً ، أي مقسماً إلى قطع (زجل module) مستقلة . هذا ما يؤمن لنا بعض الاهتمامات : تبسيط البرامج وتنقيص طول المهام ، إعطاء البرنامج كاملاً تركيبة زجلية تسمح بتسهيل عملية تعديل البرنامج ، تسهيل عمل الفريق (العمل الجماعي) ... ونحصل على ذلك بتقسيم البرنامج إلى عدة أقسام - مصدر ، باستعمال الإمكانات التي تضعها البرامج الثانوية بتصرفنا (أنظر الفصل 21) ، وباستعمال أوامر (توجيهات) التقسيم .

قسم مهم من عمل المؤول يقوم على ربط الرموز الموجودة في الزجل (الأقسام) بعناوين محددة على شكل قاعدة ، مؤشر وإزاحة . ينتهي المؤول من العمل عندما يلتقي الأمر END الذي يشير إلى نهاية الزجلة . تتألف الزجلة المصدر من مجموعة من التعليقات المؤولة في مرة واحدة .

1.2.20 . رموز داخلية ، رموز خارجية

يمكن تصنيف الرموز التي يلتقيها المؤول في زجلة مصدرية ، في عدة طبقات .

1 - الرموز المطلقة .

2 - الرموز المنقولة التي تظهر في منطقة الوسم . وهي تسمح عادة ببلوغ تعليمة أو معطى ما . ولا يمكنها أن تظهر إلا مرة واحدة في منطقة الوسم خوفاً من التعريف المزدوج . كما أنها داخلية ضمن زجلة المنبع ويقوم المؤول بربطها بعنوان على شكل قاعدة وإزاحة . ويقوم بتخزينها في جدول الرموز المنقولة (المترجمة) .

3 - الرموز التي تظهر في منطقة الوسم ولكن من النوع « نقاط الدخول » . وتنتمي إلى زجلة المصدر ولكنها قد تكون قابلة للتسمية بواسطة أسماء من خارج هذه الزجلة . من الممكن تصنيفها في طبقتين : طبقة الرموز المستعملة . في تسمية التعليقات ، وطبقة تلك التي تستعمل لتسمية مناطق المعطيات . يقوم المؤول بتخزينها في جدول

الرموز الخارجية ESD (External Symbol Dictionary) حتى لو كانت داخلية في زجلة المصدر . رمز واحد على الأقل ينتمي إلى الفئة الأولى : الرمز الذي يشير إلى التعليمات الأولى للتنفيذ . إذا كان هذا الأمر غائباً فإن المؤول يختار كنقطة دخول عنوان التعليمات الأولى من البرنامج ويخزّنه في ESD . يجب تعداد الرموز من النوع نقاط الدخول في الأمر ... ENTRY SYMB1, SYMB2, ... إذا لم تكن معتبرة كنقاط دخول إذا كانت مستعملة لتسمية القطعة (الزجلة) .

4 - الرموز التي تظهر في زجلة منطقة العوامل ولكن غير الموجودة في منطقة الوسم . هذه الرموز تنتمي إلى زجل مصدريّة أخرى ولا يستطيع المؤول أن يربط عنواناً بها ؛ وهو يعهد بهذه المهمة إلى مُنقَح الأربطة (link editor) أو إلى الشاحن ، وذلك بتخزينها في ESD . تُعتبر هذه الرموز خارجية بالنسبة لزجلة المصدر . إنّها عبارة عن نقاط دخول إلى زجل أخرى وإذا فهي تنتمي إلى إحدى الطبقتين المذكورتين في 3 . ويجب أن يكون مصرحاً عنها وكأنها خارجية بواسطة الأمر , EXTRN SYMB1 SYMB2 ... إذا كانت عبارة عن أسماء برامج ثانوية مصرحاً عنها في ثابتة بعنوان من النوع V .

2.2.20 . أوامر التقسيم

هذه الأوامر تشير إلى بداية أو نهاية قسم من زجلة المصدر .

[تعبير منقول (مترجم)] END

يشير إلى نهاية زجلة المصدر . العنوان المناسب للتعبير المنقول يُخزّن في ESD .
إنّه بشكل عام عنوان أول تعليمات للتنفيذ .

```

CSECT
-----
ALPHA
-----
-----
END      ALPHA

```

يُعرف ALPHA كنقطة دخول إلى البرنامج .

قسم التحكم (Control section) هو عبارة عن قطعة منقولة من البرنامج (قابلة للترجمة) . هذا يعني بأنّه يجب أن نربطها برصف قاعدة واحداً على الأقل ، مما يجعل هذه الوحدة قابلة للنقل والترجمة بشكل مستقل عن باقي البرامج . وهي تبدأ بحدود كلمة مزدوجة . يمتد قسم التحكم من بداية القسم حتى إلتقاء قسم آخر .

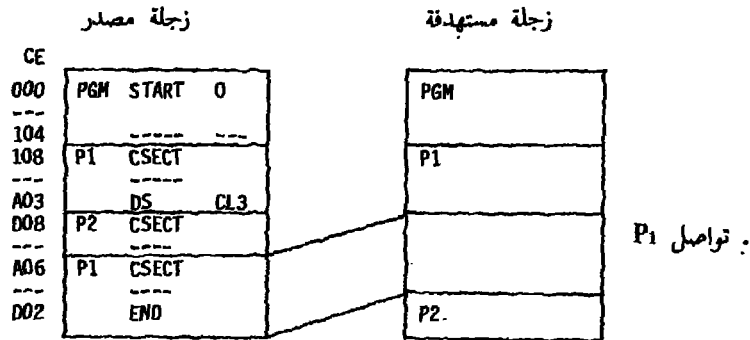
[symbole] START [constante]

[ثابتة] START [رمز]

يقوم بإعداد قسم التحكم الأول بـزجلة المصدر . الثابتة الاختيارية تسمح بإعطاء قيمة أولية إلى عداد المواقع . يُخزّن الرمز في ESD .

[Symbol] CSECT

يعرّف عن قسم التحكم أو مؤشر إلى قسم داخلي . الإلتقاء الأول للرمز يشير إلى بداية القسم ، والإلتقاء التالي لنفس الرمز يشير إلى مواصلة القسم . يعمل المؤول قسماً بعد قسم : مختلف قطع القسم تكون موجودة متصلة في نفس الزجلة المستهدفة (object module) ، هكذا في المثل التالي ، يتم تأويل تواصل P1 قبل P2 . من هنا نحصل على قاعدة كيفية تطوّر CE .



يُخزّن الرموز PGM ، P1 و P2 في الجدول ESD . وهي تُمثل نقاط الدخول . تشير إلى أن جميع أقسام التحكم يجب أن تعرّف بواسطة رمز ما عدا واحداً . يمكن أن يُعرّف بواسطة اسم أبيض . يجب على كل قسم ، وهذا موجود في التعريف ، أن يتمتع بمصرف قاعدة . ويُعرّف المؤول العناوين الفيزيائية للقسم باستعمال هذا المصرف القاعدي الذي يجب أن يُشحن مع قيمة العنوان المناسب . يمكن لقسم التحكم أن يبدأ على الشكل التالي :

[symbole] CSECT (أو START للأولى)
BALR RBASE,0 (حيث RBASE هو المصرف القاعدي)
USING *,RBASE

سنعرض عليكم حلاً آخر لشحن المصرف القاعدي في الفصل 21 .
القسم الوهمي (dummy section) هو عبارة عن قسم مستعمل فقط لوصف المعطيات دون حجز لمواقع لها في الذاكرة وسمح إذاً بتعريف رموز دون ربطها بعناوين في لحظة كتابة القسم الوهمي . المثل التالي سيوضح ذلك :

لنفترض البرنامج التالي الذي يستعمل المنطقتين Z1 و Z2 المنفصلتين فيزيائياً مع أنها بتركيبة متشابهة . سنقوم بتعريف التركيبة المشتركة في تركيبة وهمية تدعى ENREG وستطبقها على Z1 و Z2 عندما يصبح ذلك ضرورياً .

Z1	DS	CL80	
Z2	DS	CL80	حجز المناطق
	---	---	
	USING	ENREG,4	تعريف العنوان بالنسبة للقسم الوهمي
	L	4,=A(Z1)	
	---	---	
			تطبق تركيبة القسم الوهمي على Z1
	L	4,=A(Z2)	
	---	---	
			تطبق تركيبة القسم الوهمي على Z2
ENREG	DSECT		
NUMERO	DS	CL4	
MONTANT	DS	CL10	
NOM	DS	CL20	
ADRESSE	DS	CL46	

[symbole] DSECT

يُعرف عن بداية أو تواصل القسم الوهمي . عنوان القسم يمكن أن تتم بفضل وجود الرمز الموجود قبل DSECT أو بفضل وجود أي رمز في الوصف . يُوضع عدّاد الرموز دائماً في صفر عند بداية DSECT . يُحزّن الرمز في ESD . من هنا نلاحظ البساطة الناتجة عن هذا المفهوم . والبرمجة ستكون مُبسّطة ومن هنا ينتج إقتصاد في استعمال الرموز .

القسم المشترك يسمح لعدة زجل مصدر ، مؤولة بشكل منفصل ولكن متّحدة فيما بينها بواسطة منقح الأربطة ، أن تتقاسم نفس منطقة التنفيذ . سنستعمل هذه المنطقة :

- لإيصال المعطيات بين زجل المصدر (فورتران ومؤول مثلاً) ،
 - كمنطقة عمل مؤقتة لإحدى الزجل بشرط ألا تستعمل في نفس الوقت .
- عند المعالجة بالمؤول سيتم حجز موقع لكل زجلة ، ولكن عند المعالجة بواسطة مُنقّح الأربطة فإن المناطق المشتركة ستتحّد ، وفقط ستحفظ المنطقة ذات الحجم الأكبر .

[Symbol] [رمز] COM

تُعرف عن منطقة مشتركة . يسمح النظام OS بوسم المناطق ولكن النظام DOS لا يسمح بذلك (لا يوجد رموز) . من الضروري ، في كل زجلة مصدر ، أن يتم

إجراء عنوانه بشكل شبيه بما جرى في DSECT . يوضع عداد المراكز في صفر عند بداية القسم .

3.2.20 . تنقيح الأربطة (link edition)

الفقرات السابقة تسمح لنا بفهم ويشكل أفضل عمل مُنقَّح الأربطة والشاحن (loader) .

مع الزجلة المستهدفة ، يقدم المؤول إلى مُنقَّح الأربطة جدولاً ESD لكل زجلة مصدر . نجد في الجدول ESD أسماء الرموز من الفئتين 3 و4 (فقرة 2.2.20) . في كل رمز نجد كود العملية من نوع الأمر المرتبط بها . إذا كان الرمز من نوع نقطة الدخول ، فإن عنوانه هو في الزجلة المشار إليها . بالنسبة للزجلة المصدر المذكورة في الفقرة 4.2.20 ، فإن الجدول ESD يكون على الشكل التالي :

EXTERNAL SYMBOL DICTIONARY

SYMBOL	TYPE	ID	ADDR	LENGTH	LDID
	PC	0001	000000	00001C	
ALPHA	ER	0002			
PI	SD	0003	000020	00000C	
DEBUT	ER	0004			
SP	ER	0005			

يكوّد نوع الرمز على الشكل التالي :

كود	متاسب للأمر
PC	بدون وسم START ou CSECT
SD	مع وسم START ou CSECT
DM	COM
XD	(1) خارجي، DXD ou DSECT
LD	ENTRY
ER	أو ثابتة بعنوان EXTRN DC V(...)
WX	WXTRN (2)

في مقابل هذه المعلومات المرتبطة بكل زجلة ، فإن مُنقَّح الأربطة يقوم بالإجابة على الطلبات الخارجية ، أي يقوم بإجراء التناسب بين الأسماء الموجودة في مختلف ESD . وإذا لم يكن بإمكان المُنقَّح أن يحل مشكلة الطلبات الخارجية بسبب جدول الزجل ESD المطلوب ربطها ، فهو يقوم بعملية بحث منتظمة في المكتبات التي يقدر على بلوغها .

(1) CXD ، DXD ، DSECT الخارجية هي غير مشروحة في هذا الكتاب .
(2) WXTRN تقوم بملء نفس الدور الخاص بـ EXTRN . في ما يتعلّق بالساح لمنقَّح الأربطة بالبحث الأتوماتيكي عن الرموز بداخل المكتبة ، فإن WXTRN تمنع هذا البحث .

4.2.20 . الشحن (loading)

يقوم الشحن على تخزين البرنامج في الذاكرة بدءاً من عنوان مُحدّد . كما رأينا في الفقرة 2.3 ، العناوين المنقولة لا يجب أن تتعدّل خلال هذه العملية . والأمر ليس كذلك بالنسبة لثوابت العناوين . يقوم الشاحن بتخزين العناوين الفعلية للمتأثرات المطلوبة في الذاكرة .

يجب على المؤول أن يرسل إلى الشاحن مواقع المناطق المطلوب إعادة حسابها . يستعمل لهذا الهدف RLD (Relocation Dictionary) حيث تتواجد عناوين ثوابت العنوان . الجدول ESD في المثل أعلاه هو موجود في الفقرة 3.2.20 . نذكّر بأن DC V (SYMB) يعادل :

EXTRN SYMB

DC A(SYMB)

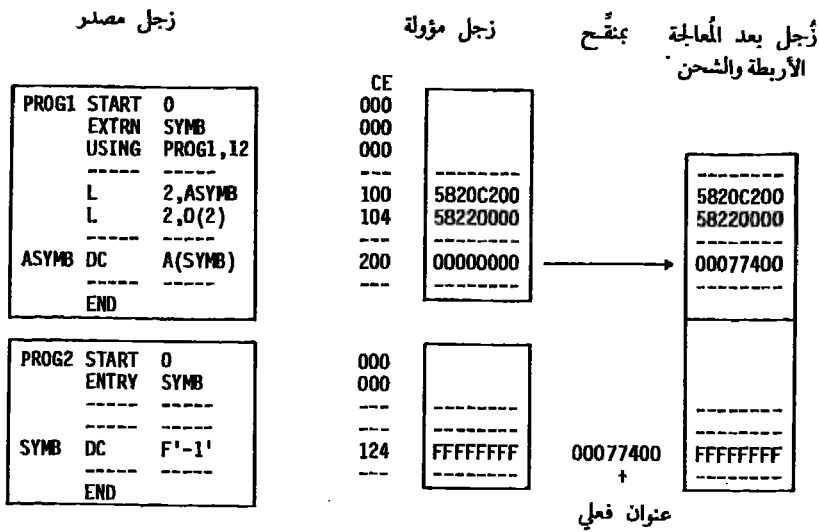
يُحتفظ باستعمال ثوابت العنوان من النوع V للتعريف عن عنوان تفريع (إسم قسم ، إسم برنامج ثانوي . . .) الرمز SYMB يُخزّن في ESD . ويقوم المؤول بتصفير الثابتة .

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE	STATEMENT
000000				1		START 0
			00000	2		USING *.12
				3		EXTRN ALPHA
000000	5830 C010	00010		4	DEBUT	L 3.=A(ALPHA)
000004	5850 C014	00014		5		L 5.=A(BETA)
000008	FFFFFFFF			6	ZONE	DC F'-1'
000020				7	PI	CSECT
			00020	8		USING *.11
000020	5850 C018	00018		9	BETA	L 5.=V(SP)
000024	00000000			10	ADR	DC A(DEBUT)
000028	00000000			11		DC V(DEBUT)
				12		END
000010	00000000			13		=A(ALPHA)
000014	00000020			14		=A(BETA)
000018	00000000			15		=V(SP)

RELOCATION DICTIONARY

POS.ID	REL.ID	FLAG5	ADDRESS
0001	0002	0C	000010
0001	0003	0C	000014
0001	0005	1C	000018
0003	0001	0C	000024
0003	0004	1C	000028

سنفحص في المخطط التالي كيفية تطوّر القيمة المأخوذة من قبل ثابتة عنوان من التأويل إلى الشحن :



5.2.20 . الاتصال بين أقسام نفس الزجلة المصدر لنأخذ المثال التالي :

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000000				1 P1	CSECT
				2 *	---
		00000		3	USING P1,12
				4 *	---
000000	0000 0000	00000		5	L 3,SYMB2
	*** ERROR ***				
000004	5840 C010	00010		6 *	---
000008	5844 0000	00000		7	L 4,=A(SYMB2)
				8	L 4,0(4)
00000C	00000001			9 *	---
				10 SYMB1	DC F'-1'
				11 *	---
00001E				12 P2	CSECT
				13 *	---
		00018		14	USING P2,11
				15 *	---
000018	5830 C00C	0000C		16	L 3,SYMB1
				17 *	---
00001C	FFFFFFFF			18 SYMB2	DC F'-1'
				19 *	---
000010	0000001C			20	END
				21	=A(SYMB2)

ولنعرض المشاكل التي يفرضها الاتصال بين قسمين عند إجراء مرحلتين من التأويل والتنفيذ .

1 - عند التأويل فإن أي مشكلة خاصة لن تواجهنا . ينتمي القسمان إلى نفس زجلة المصدر ويمكن أن يقوم المؤول بإجراء شروط العنونة لتجميع الرموز الداخلية بشرط أن توافق القواعد العائدة إلى USING . هكذا ، فتأويل السطر الخامس لا يمكن أن يتم لأن هذه التعليمة لا تنتمي أبداً إلى حقل USING P2,11 . في القسم P1 ،

نستطيع بلوغ SYMB2 باستعمال ثابتة العنوان A(SYMB2) التي يقوم الشاحن بإعدادها بشكل مناسب . وفي المقابل ، فإن التعليمات SYMB1 16, 3, L يمكن أن تكون مؤولة .

2- عند التنفيذ ، تكون المشكلة مختلفة : التعليمات SYMB1 3, L هل ستسمح بالبلوغ إلى SYMB1 ؟

قد يسمح لنا التأويل المناسب للتعليمات بهذا الافتراض . هكذا فعملياً هذه التعليمات تسمح عند التنفيذ ، ببلوغ SYMB1 بشرط أن تكون القاعدة 12 المُنونة SYMB1 تحتوي على العنوان P1 المناسب . ولكن لا شيء مؤكداً ، في مثل معاكس ، يكفي أن يكون القسم P2 مُنفذاً قبل القسم P1 كي لا تكون القاعدة 12 مشحونة بشكل مناسب . إضافة لذلك ، فإن أي مراجعة من هذه الطبيعة تناقض تعريف قسم التحكم . وبالتالي فإننا سنراجع SYMB1 في P2 بفضل وجود ثابتة العنوان .

يظهر إذاً وبوضوح أنّ الأقسام يجب أن تُعتبر كوحدات مُستقلة في نفس الوقت الذي تكون فيه الزجل المصدرية منفصلة عند التأويل . الاتصال الرمزي بين الأقسام سيتم دائماً بواسطة ثوابت العنوان . هذه التقنية تسمح بتفادي العقبة المثارة أعلاه وتسمح بدون مشكلة بتوزيع الأقسام في مختلف زجل المصدر .
وبإيجاز ، فإن تفريع القسم سيتم بواسطة :

$$\begin{matrix} L & R,=V(P1) & (= A (P1)) \\ BR & R & \text{حيث } R \text{ هو مصرف عام} \end{matrix}$$

R هو مصرف عام ، بشكل عام المصرف 15 حسب إتفاقات الربط المعروضة في الفقرة 4.21 .

بلوغ الرمز يتم بواسطة :

$$\begin{matrix} L & R,=A(SYMB) \\ L & R,0(R) \end{matrix}$$

6.2.20 . ختام حول التقسيم

يعطي التقسيم وسيلة لتجزئة زجلة المصدر إلى زجل مُستقلة . عند إجراء التقسيم فإن كل شيء يجري كما لو كانت الزجل المصدرية مترابطة .

نحرص على عدم بلوغ ، في نفس القسم ، رموز لا تنتمي إلى هذا القسم . وإذا كنا نرغب ببلوغ رموز خارجية فسنستعمل الطريقة المعروضة في الفقرة 4.2.20 ، تاركين إلى الشاحن مهمة إجراء الوصلة بواسطة ثوابت العنوان .

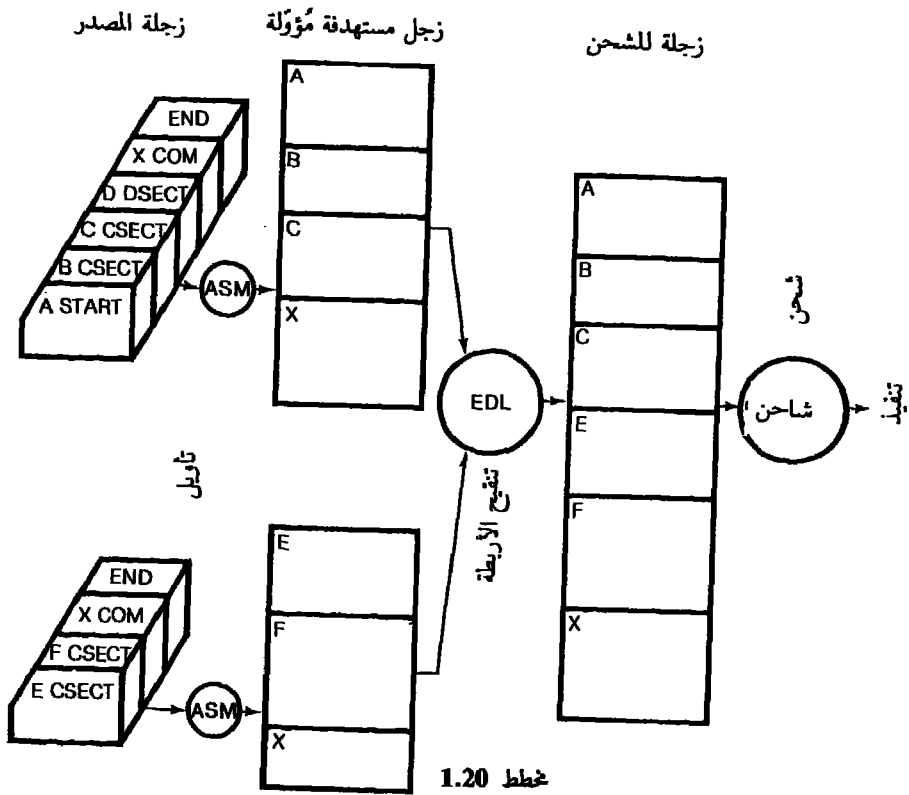
يجب على كل قسم أن يحتوي على مرصف قاعدة ، ويجب شحن هذا المرصف ، في لحظة التنفيذ ، بالعنوان المناسب . ستجري دراسة هذه المسألة في الفصل التالي .

بعد أخذ هذه الاحتياطات بعين الاعتبار ، فإن التقسيم يؤدي إلى تحسين كبير في تنظيم المعالجة بالمؤول . وهو يسمح ، عند الحاجة ، « بتفتيت » وبدون مشكلة البرنامج إلى زجل دون أي خوف على الترابط العام .

وبشكل عام فإن الأقسام هي برامج ثانوية . يجب إذن الاعتناء ، عند الدخول إلى قسم من هذا النوع ، بتخزين مرصيف البرنامج المؤادي .

ويُعالج الفصل 21 هذه المشكلة . لا يجب الخلط بين القسم والبرنامج الثانوي اللذين يمثلان مفهومين مختلفين . من الممكن القول أن تقسيم البرنامج هو عبارة عن نقل قسم من العمل الجاري بواسطة المؤول إلى مُنقَح الأربطة والشاحن .

سنلاحظ في المخطط التالي إختفاء DSECT من الزجلة المؤولة والموقع الوحيد المشغول بواسطة COM في الزجلة المشحونة . المكان المشغول بواسطة القسم المشترك يُعادل الحجم الأكبر بين الاثنين .



3.20 . الأوامر التي تُغيّر عدّاد المواقع
 ORG عبارة عن تعبير منقول أو مطلق . هذا الأمر يؤدي إلى تغيير الازدياد الطبيعي لعدّاد المواقع . وهو يسمح بشكل خاص بإجراء إعادة تعريف أو حجز مكان من الذاكرة . إذا كانت منطقة العناصر (القياسات) فارغة ، فإن ORG يعطي عدّاد المواقع CE القيمة التي كانت موجودة فيه عند آخر تعديل بواسطة ORG . لا يمكن أن يكون القياس (argument) مبلوغةً في البداية .

قيمة العداد
 CE

OCO	TABLE	DC	XL256'40'
OCA		ORG	TABLE+10

ODO		ORG	

LTORG عبارة عن أمر بدون قياسات . وهو يشير إلى المكان الذي يجب أن تُؤوّل فيه الثوابت الحرفية . في غياب هذا الأمر فإن تأويلها سيتم في نهاية أول قسم .
 CNOP b, w يؤدي ، بحكم عدم إجراء أية عملية ، إلى زيادة قيمة عدّاد المواقع إلى الحد الأقرب لنصف كلمة ، كلمة أو كلمة مزدوجة حسب قيمتين b و w .

CNOP	0,4	بداية كلمة
CNOP	2,4	وسط كلمة
CNOP	0,8	بداية كلمة مزدوجة
CNOP	2,8	النصف كلمة الثاني من كلمة مزدوجة
CNOP	4,8	النصف كلمة الثالث من كلمة مزدوجة
CNOP	6,8	النصف كلمة الرابع من كلمة مزدوجة

4.20 . أوامر التحكم باللوائح
 ICTL يسمح بتعديل الإطار النموذجي (الأعمدة 1 ، 16 و 71) للتعليقات .
 ISEQ يسمح بالتحقق من الترتيب المتتالي للبطاقات .
 COPY يسمح بنسخ قسم من النص المصدر في المكتبة .
 EJECT يؤدي إلى ظهور التعليمة التالية في رأس الصفحة التالية من اللائحة .
 وهو مفيد لتوضيح نص البرنامج .
 SPACE n يسمح بإدخال عدد n من الأسطر الفارغة في اللائحة .

PRINT [ON, GEN, NODATA]
 [OFF, NOGEN, DATA]

يسمح بالمحافظة على أو بإلغاء اللائحة (Listing) ، توليد الماكرو تعليقات توليد المعطيات .

«سلسلة» TITLE يسمح بطباعة عنوان من 100 سمة في رأس كل صفحة .
PUNCH, REPRO يسمحان بثقيب البطاقات .

5.20 . أوامر مُستعملة بإشراف النظام OS فقط
OPSYN يسمح بتعريف مجموعة كود العمليات الخاصة المرادفة للأكواد IBM .
هذا الأمر يمكن أن يكون مفيداً بشكل خاص لاستبدال كود - عملية خاص بمأكرو
عملية .

من الممكن إذاً تبديل الكود الحرفي BE ، BNE ، للماكرو حيث الأسماء
سيصرّح عنها بشكل مرادف بسبب وجود OPSYN . هذه الماكرو تعليمات تولّد كلمة
تُخزّن فيها نتيجة الاختبار الذي يسبق تعليمة التفريع بالشكل V أو F أو O أو N « وبعد
ذلك تقوم بالتفريع المناسب باستعمال التعليمة BC أو BCR . هذه السات V أو F
ستكون مرثية في العملية DUMP (دلق) وتسمح بمتابعة أثر تنفيذ البرنامج (Trace) .
بالإمكان تمييز مختلف الأسماء المُولدة بواسطة SYSNDX & (فقرة 7.2.22) .

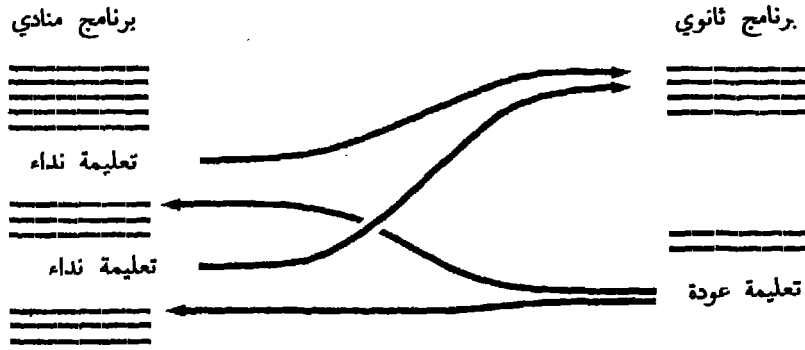
بعد مرحلة الإطلاق في العمل ، فإن إلغاء الأوامر (التوجيهات) OPSYN يؤدي
إلى تفادي إدخال ماكرو التعليمات والبدء بتنفيذها .
من الممكن أيضاً استعمال هذا الأمر لجعل بعض التعليمات غير عملية وذلك
بجعلها مرادفة للتعليمة NOP (لا عملية) .

PUSH و POP . من الممكن عند كتابة البرنامج أن نقوم بشحن مصرف القاعدة
بسرعة وأن نستعيد القاعدة القديمة لاحقاً . هذا يمكن أن يتم مثلاً ، عندما تستعمل
إحدى ماكرو التعليمات قاعدة شخصية . بعد التبديل ، بواسطة المؤول ، يجري فقدان
القاعدة القديمة . يسمح الأمر PUSH بتخزين المرافف وعنوان القاعدة وصيغ الأمر
PRINT داخل مكّس (Stack) ⁽¹⁾ . POP يُعاود إسترجاع المفهوم القديم بواسطة
إستخراج لآخر كلمة مكّسة .

(1) المكّس هو عبارة عن جدول مُنظّم حسب التقنية «الداخل أخيراً هو الخارج أولاً» .

21 البرامج الثانوية

البرنامج الثانوي هو عبارة عن سلسلة من التعليقات التي يتم تنفيذها بطلب من تعليمة نداء (Call) . عندما ينتهي تنفيذ البرنامج الثانوي يعود العمل بالبرنامج المُنَادِي. وبالتعليمة التي تتبع مباشرة تعليمة النداء . المخطط التالي يوضح هذه الأولوية :



كل شيء يجري كما لو كانت تعليقات البرنامج الثانوي داخلة في مكان تعليمة النداء .

بإمكاننا تقسيم البرنامج الى مهام (task) ، كل مهمة يتم حلها بواسطة برنامج ثانوي . إعداد البرنامج بكامله يصبح سهلاً ، والأقسام تصبح صغيرة . هذه الأولوية تسمح بتقادي إعادة كتابة التعليقات المتشابهة عندما يجب تنفيذ البرنامج في مختلف مستويات البرنامج المُنَادِي . وتطرح هذه التقنية مشكلتين :

- تخزين عنوان العودة (العنوان الذي يتبع مباشرة عنوان تعليمة المُناداة) ،
- إنتقال المتغيرات الوسيطة .

مشكلة إنتقال المتغيرات جرت إثارتها في إطار تقسيم البرنامج ولكن البرنامج الثانوي لا يُشكّل بالضرورة قسم تحكم

1.21 - البرنامج الثانوي وقسم التحكم

التقسيم هو عبارة عن عملية تتعلق بالتأويل ، تنقيح الأربطة والشحن : أما مفهوم البرنامج الثانوي فلا يتعلق سوى بالتنفيذ . مناداة البرنامج الثانوي تؤدي ، عند التنفيذ ، إلى تعديل الدوران المتالي للتعليقات .

هكذا ، فلا شيء يعترض بأن يكون البرنامج والبرنامج الثانوي تابعين لنفس القسم . ولكن هذا النوع من التنظيم لا يُقدّم جميع الفوائد التي ننتظرها من البرنامج الثانوي . فهو يربط البرنامج بالبرنامج الثانوي بينما نرغب نحن بجعل البرنامج الثانوي قابلاً للطلب والدعوة من جميع الأقسام أو الزجل . وهو لا يشكل تحسّيناً باتجاه تركيبة زجلية . وبالتالي لا يستعمل إلا عندما يكون البرنامج الثانوي مرتبطاً بشكل كبير منطقياً بالبرنامج المُنادي .

في أغلب الأحيان يُفضّل إستعمال إمكانيات التقسيم : سيشكل البرنامج الثانوي قسماً من البرنامج .. من المحتمل ، منذ لحظة تصوّر البرنامج الثانوي ، إستعمال هذه الزجلة في مُعالجات أخرى . يُفضل معالجة مشكلة الاتصال بين البرنامج / البرنامج الثانوي كوصلة ببرنامج خارجي تسمح بإمكانية تفكيك عمليات التأويل دون تعديل في الأقسام .

2.21 . تفريع إلى برنامج ثانوي والعودة

مناداة البرنامج الثانوي ليست سوى قطع إلزامي للدوران المتالي للتعليقات ولكن مع تخزين للعنوان التالي الذي يتبع تعليمة المناداة بشكل يسمح بمعاودة العمل بالبرنامج المقطوع . تتمتع كل مكنة بأولية خاصة للتفريع مع عودة . يستعمل النظام 360/370 التعليمتين BAL و BALR اللتين رأيناهما في الفصل 12 .

BAL R1,D2(X2,B2)
BALR R1,R2

يكون عنوان العودة مُخزّناً في المرصف R1 . يكفي إذاً في نهاية البرنامج الثانوي أن نشحن عدّاد البرنامج بالقيمة المخزّنة في R1 بواسطة التعليمة BCR 15,R1 مثلاً . نحصل إذاً على التركيبة التالية :

البرنامج المُنادي

البرنامج الثانوي SP

L
BALR

SP -----

(تخزين المرصيف وتعريف القاعدة)

R2,=A(SP) (V(SP) أو)
R1,R2 خارجياً

(إعادة مضمون المراضف إلى الذاكرة)
BCR 15,R1

إذا كانت BALR موجودة على العنوان ALPHA ، فإن BCR.15,R1 تعيد تخزين
ALPHA +2 في عداد البرنامج (CO) .
كان بإمكاننا إستعمال BAL بأحد الأشكال التالية :

- 1°) BAL R1,SP عبارة عن مرجع داخلي
2°) L R2,=A(SP) ou =V(SP)
BAL R1,DEPLAC(R2)

الشكل الذي يسمح ، بواسطة حساب بسيط لـ DEPLAC ، بالحصول على
مداخل متعددة في SP .

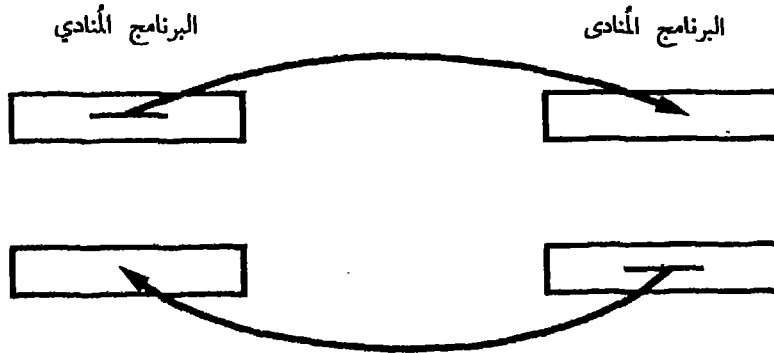
نلاحظ أنه لا يوجد فرق أساسي بين التفرعات إلى برامج ثانوية خارجية أو
داخلية . وحده تعريف ثابتة العنوان الخارجي هو إلزامي في الحالة الأولى .

3.21 . إنتقال المتغيرات الوسيطة

المشكلة الثانية في عملية الاتصال بين البرنامج والبرنامج الثانوي تكمن في عملية
تبادل المعطيات . إن تقنيات عبور المتغيرات هي متعددة ويمكن للقارئ أن يتصور
الطريقة الأفضل لمسأله . ولكن من المفيد هنا أن نعرض الطرق العامة التي تساعد على
الاختيار . تُستعمل اللغات المتطورة بطريقتين أساسيتين : لانتقال المتغيرات مباشرة بالقيم
والانتقال بالعناوين .

إنتقال المتغيرات حسب القيم

ويمكن في نسخ القيمة المطلوب إرسالها إلى منطقة معروفة من البرنامج المنادي .



هذه المنطقة يمكن أن تكون خلية في الذاكرة مركزية (Local) في البرنامج المُنادى أو مرصفاً. تستعمل هذه التقنية ، مثلاً في لغة فورتران ، لاعادة قيمة إحدى الدوال إلى البرنامج المُنادي . وبشكل عام فإن النتيجة تُخزن في المرصف 0 بواسطة البرنامج المُنادي .

نلاحظ إنه إذا كانت B عبارة عن متحوّلة مركزية من البرنامج المُنادى ، فإن أي تعديل في B لا يؤدي إلى أي تغيير في الخلية A .

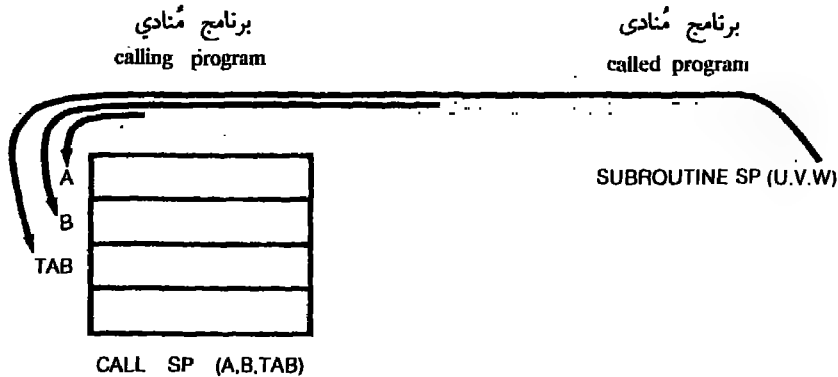
وفي لغة المؤول ، يمكن أن تُحلّ مشكلة التبادل بالقيم بواسطة النقل بالمراسف ، حيث يُحدّد المبرمج طريقة لاستعمال المراسف .

البرنامج المُنادي	البرنامج المُنادى
-----	-----
L 1,A	ST 1,U
L 2,B	ST 2,V
SP 3u SP	-----
تفريع إلى SP	

نشير إلى أن هذه الأوامر هي غير متوافقة مع تبادل الجداول . فعندئذٍ تتطلّب مكاناً كبيراً من الذاكرة . هذه الطريقة هي غير مناسبة إلا عندما يكون عدد المعطيات المطلوب إرسالها قليلاً .

إنتقال المتغيرات بواسطة العناوين

وتكمن هذه الطريقة بإرسال عناوين المتغيرات إلى البرنامج المُنادى . يعمل البرنامج المُنادى إذاً على معطيات البرنامج المُنادي . يبلغ البرنامج المُنادي قيم المتغيرات بواسطة العنونة غير المباشرة . أي تعديل ، في البرنامج المُنادي ، في قيمة منقولة ، معناه تعديل منطقة من البرنامج المُنادي . هذه الطريقة هي نفسها المُستعملة للإرسال بواسطة CALL (Call SP name, arguments list) في فورتران . المخطط التالي يوضح لنا عملياً كيف أن متحوّلات البرنامج المُنادي تصبح مركزية في البرنامج المُنادي .



تُدعى متغيرات وهمية الرموز A ، B ، TAB الواردة في تعليمة النداء لأنها تتمتع فعلياً بقيمة معينة في لحظة النداء أو عند العودة .

تُدعى متغيرات شكلية الرموز U ، V ، W من SP التي ليست سوى أسماء تمثل ، في لحظة النداء ، الرموز A ، B ، TAB من البرنامج المُنادي .

في لغة المؤلف بإمكان المبرمج تصوّر عدة حلول لنقل المتغيرات إلى البرنامج المركزي ، فلنذكر البعض منها .

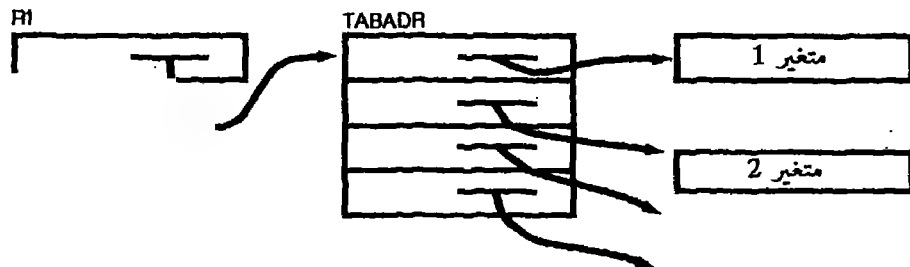
1 - نضع المتغيرات في الجدول TAB ونرسل عنوان الجدول بواسطة أحد المرافف .

نداء	برنامج ثانوي
L R1,=A(TAB)	يتم بلوغ المتغير n بواسطة
L R15,=A(SP)	L R4,DEPLAC(R1)
BALR R14,R15	أو بالتأشير
	L R4,0(R5,R1)
	وعندئذ يوضع المتغير بتصرفه في R4

2 - نضع عنوان الجدول TAB مباشرة بعد تعليمة النداء

النداء	برنامج ثانوي
L R15,=A(SP)	R14 يسمح ببلوغ TAB .
CNOP 2,4 (تراصف)	العودة تتم بواسطة :
BALR R14,R15	BC 15,4(R14)
DC A(TAB)	

3 - تكون المتغيرات عادة غير مترابطة في البرنامج ونُفضّل عادة اعتماد التقنية المستعملة بواسطة المصرفات . نقوم بإرسال عنوان الجدول الذي يحتوي على عناوين المتغيرات بواسطة أحد المرافف .



نداء		برنامج ثانوي	
L	R1,=A(TABADR)	WORK EQU ...	مرصف عمل
L	R15,=A(SP)		
BALR	R14,R15	L WORK,0(R1)	
		L WORK,0(WORK)	
			المتغير الأول في WORK
		L WORK,4(R1)	
		L WORK,0(WORK)	
			المتغير الثاني في WORK

هذا الحل هو المعتمد في لغة فورتران ، ويسمح ، في لغة الماول ، باستعادة المتغيرات المرسله بواسطة أحد البرامج فورتران وبالعكس .

نشير هنا إلى الفرق بين المتغيرات المرسله ومتغيرات العودة ، وهي تنتمي إلى البرنامج المُنَادِي . كما نفضل إستعمال مرصاف حسب نفس الاتفاقات المستعملة في أنظمة التشغيل (فقرة 4.21) . تسمح التعليمه CALL بإرسال من هذا النوع .

4.21 . إتفاقات الإتصال بين النظام والبرنامج

يبدأ التنفيذ منذ اللحظة التي يتم فيها إعداد عدّاد البرنامج وتخزين عنوان التعليمه الأولى للتنفيذ فيه . يقوم نظام التشغيل بهذه المهمة ، مما يفترض علينا إعتبار كل برنامج مستعمل كبرنامج ثانوي للنظام . من هنا فإن برنامج المستعمل يجب أن يبدأ بتمهيد يتعلّق بشروط إستعمال المرصاف من قبل النظام .

تسمّى المرصاف 0 ، 1 ، 13 ، 14 و 15 مرصاف ربط «linkage registers» في وثائق المُصنِّم . وتستعمل بواسطة النظام والمصرفات بشكل نموذجي وهذا هو السبب الذي من أجله يعتمد المستعمل على نفس الاتفاقات في الاتصالات مع البرامج الثانوية الخاصة به . في النظام OS ، يجب على البرنامج الثانوي أن يحمي مرصاف المُنَادِي في منطقة تدعى SAVE AREA ، تنتمي إلى البرنامج المُنَادِي . تجلّد تركيبة هذه المنطقة على الشكل التالي :

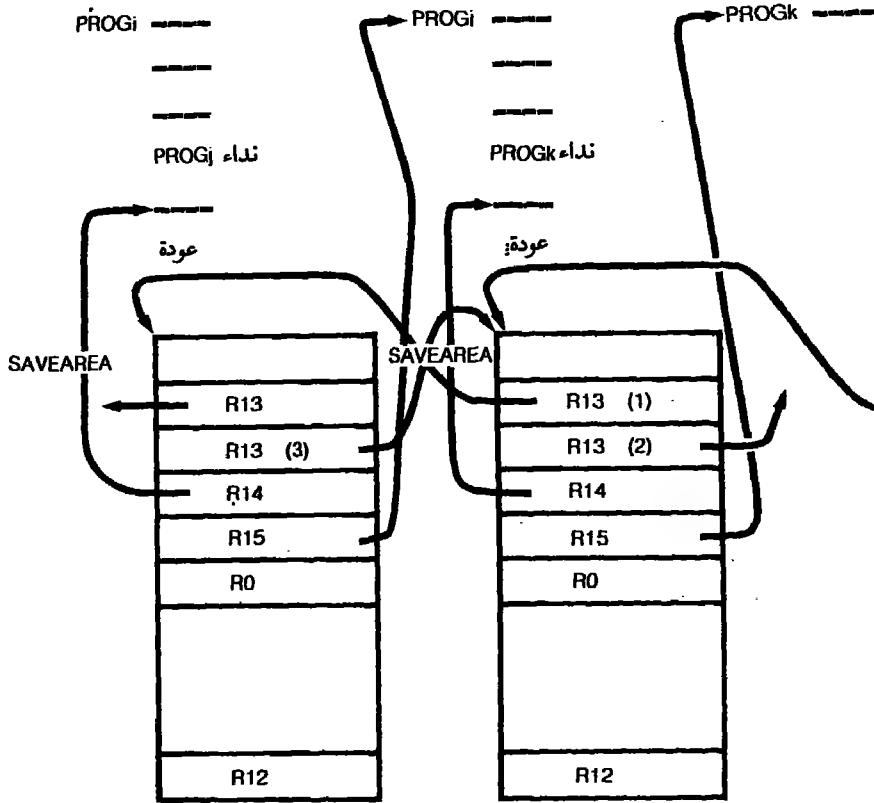
الكلمة المحتوى

- 1 تستعمل بواسطة اللغة PL/1
- 2 عنوان SAVE AREA الداخلي السابقة (الخاصة بالمُنَادِي) .
- 3 عنوان SAVE AREA التالية (الخاصة بالمُنَادِي) .
- 4 عنوان العودة إلى المُنَادِي (مرصف 14) .
- 5 عنوان نقطة الدخول إلى البرنامج (مرصف 15) .
- 6 مرصف 0 .
- 7 مرصف 1 .

18 مرصف 12 .

- عندما ينتقل النظام التحكم إلى البرنامج :
- يحتوي المرصف 15 على عنوان نقطة الدخول إلى البرنامج . بإمكان البرنامج المُنادى أن يشحن المرصف القاعدي الخاص به بواسطة التعليمة 15, LR REGBASE ، باعتبار نقطة الدخول وكأنها عنوان قاعدي .
 - المرصف 14 يحتوي على عنوان العودة .
 - المرصف 13 يحتوي على العنوان SAVEAREA للبرنامج المُنادي . نجد هنا شرح إستعمال القاعدة 13 في التعليمة (13) STM 14,12,12 الموجودة في جميع التمهيدات للبرامج .
 - المرصف 1 يحتوي على عنوان جدول الكلمات التي تحتوي على عناوين المتغيرات الوهمية المنقولة . هذا الإتفاق يُستعمل ، مثلاً ، عندما يطلب برنامج فورتران برنامجاً آخر بلغة المؤول .
 - المرصف 0 ، يستعمل ، عند العودة ، لإرسال نتيجة إحدى الدوال (مثلاً الدالة FUNCTION في فورتران) .
- وبالنتيجة ، ومنذ اللحظة التي يأخذ فيها البرنامج المُنادى التحكم ويعود إلى التنفيذ ، فإنه :
- يُعرّف المنطقة الخاصة به SAVE AREA ،
 - يُخزن مرصيف البرنامج المُنادى بواسطة :
- STM 14, 12, 12 (13)
- في المنطقة SAVE AREA للمنادي
- يعرف مرصيف قاعدة ويشحن فيه قيمة معينة بواسطة :
- BALR , 0 أو LR , 15
- يقوم بإجراء الوصلة بين المناطق SAVE AREA : ويُخزن ، في الكلمة الثانية من المنطقة SAVE AREA الخاصة به عنوان المنطقة الخاصة بالبرنامج المُنادي (مرصيف 13) وفي الكلمة الثالثة من المنطقة SAVE AREA الخاصة بالمُنادي ، عنوان المنطقة SAVE AREA الخاصة به .
 - عند العودة ، فإن البرنامج المُنادى يعيد تخزين مرصيف البرنامج المُنادي مما يؤدي إلى العودة بواسطة BR 14 .
 - بإمكانه إستعمال المرصف 15 لترميم كود العودة .

المخطط التالي يوضح عملية الربط بين المناطق SAVE AREA .



مخطط 1.21

ملاحظات : إذا كان البرنامج المُنادى ، $PROG_i$ مثلاً ، لا ينقل التحكم إلى برامج ثانوية أخرى كالبرنامج $PROG_k$ ، فلا حاجة لتعريف SAVE AREA لهذا البرنامج . من الواجب إذا السهر على حماية المرفص 13 الذي يسمح بإعادة مفهوم التنفيذ إلى البرنامج المُنادى .

-
- (1) يتعلّق ذلك بالمرفص $R13$ من $PROG_i$
 - (2) يتعلّق ذلك بالمرفص $R13$ من $PROG_k$
 - (3) يتعلّق ذلك بالمرفص $R13$ من $PROG_j$.

LDC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE	STATEMENT
000000				1	PROGJ	CSECT OH
000000				2	PROLOGUE	DS
000000				3	*	SAUVEGARDE DES REGISTRES DE L'APPELANT
000000	90EC D00C	0000C		4	*	STM 14,12,12(13)
000000			00000	5	*	DEFINITION ET CHARGEMENT DU REGISTRE DE BASE
000000	18CF			6	*	USING PROGJ,12
000000				7	*	LR 12,15
000000	50D0 C01C	0001C		8	*	SAUVEGARDE DE R13 DANS LA SAVE AREA DE CE PROGRAMME.
000000				9	*	ST 13,SAVEAREA+4
000000				10	*	SAUVEGARDE DE L'ADRESSE DE LA SA DE CE PROGRAMME DANS LA SA
000000				11	*	DE L'APPELANT
000000	1820			12	*	LR 2,13
000000	41D0 C018	00018		13	*	LA 13,SAVEAREA
000000	50D2 0008	00008		14	*	ST 13,8(2)
000000	47F0 C060	00060		15	*	B DEBUT
000000				16	*	DEFINITION DE LA SAVE AREA
000000				17	SAVEAREA	DS 18F
000000				18	DEBUT	DS OH
000000				19	*	-----
000000				20	*	-----
000000	58F0 C070	00070		21	*	SEQUENCE D'APPEL DE PROGK
000000	05EF			22	*	L 15.=V(PROGK)
000000				23	*	BALR 14,15
000000				24	*	-----
000000				25	*	-----
000000				26	*	-----
000000				27	*	EPilogue
000000	58D0 C01C	0001C		28	*	DS SEQUENCE DE RETOUR VERS PROGJ
000000	88EC D00C	0000C		29	*	L 13,SAVEAREA+4
000000	07FE			30	*	LW 14,12,12(13)
000000				31	*	BR 14
000000				32	*	END
000000	00000000			33	*	=V(PROGK)
000000				34	*	

EXTERNAL SYMBOL DICTIONARY

SYMBOL	TYPE	ID	ADDR	LENGTH	LDID
PROGJ	SD	0001	000000	000074	
PROGK	ER	0002			

التعليمة STM تسمح بترتيب مرادف متتالية عند كل رغبة باستعمال مرادف متجاورة .

إتفاقات الربط المعرفة سابقاً تسمح بطلبات المناداة الداخلة ضمن البرامج . وهي لا تسمح أبداً بإجراء طلبات مناداة تكرارية تحتاج إلى تعريف مكدرس (STACK) خزن للنص . هذه الأوليات ليست موضوع هذا الكتاب . ولكن نشير إلى أن النظام OS يضع بتصرف المستعمل الوسائط لتعريف وإدارة منطقة من الذاكرة لكتابة برامج تكرارية (ماكرو GETMAIN) .

وللمحاجة إلى التناسق والتوافق ، فإن المبرمج سيقوم بنفس عمليات الإختيار كالنظام OS في استعمال المرادف لإجراء الوصلات بين البرامج الثانوية .

22 . التأويل المشروط وماكرو التعليمات

1.22 . التأويل المشروط

التأويل المشروط هو عبارة عن خطوة جديدة في التطور من لغة المكنة إلى اللغة المتطورة . ويتعلق ذلك بلغة تسمح بإنشاء وتوليد ، في مرحلة ما قبل التأويل ، نص مستهدف (object text) يمكن معالجته بواسطة المؤول . النص المؤول الناتج يمكن ، حسب القيم الأولية المخصصة لتحولات التأويل المشروط ، أن يتغير من تأويل إلى آخر . بإمكاننا مثلاً ، إدخال ، خلال مرحلة إعداد البرنامج ، متتالية من التعليمات (طباعة وسيطية تسمح بمتابعة أثر (trace) البرنامج) التي ، بواسطة تعديل بسيط للقيم الأولية لتحولات التأويل المشروط ، سيتم إلغاؤها عند التأويل النهائي . هذه العملية ، مضافة إلى استعمال الماكرو تعليمات⁽¹⁾ تجعل المؤول قريباً من اللغة المتطورة ، وتسمح للمبرمج بأن يُجهز بوسائل كالتعليمات : DO ... WHILE ، ... PERFORM التي تُسهّل البرمجة .

من غير الممكن هنا عرض جميع إمكانيات التأويل المشروط . سنحاول عرض الخطوط العريضة لهذه الطريقة بواسطة أمثلة توضح لنا العملية .

1.1.22 . متحولات وثوابت التأويل المشروط

التأويل المشروط يُعالج رموزاً بقيم قابلة للتغيير : وهي عبارة عن متحولات التأويل . تبدأ أسماؤها بالرمز «&» ، وتحتوي على أكثر من ثمان سيات أبجعددية ، بما فيها «&» . السمة الثانية يجب أن تكون حرفاً . متحولات التأويل هي من ثلاثة أنواع A ، B و C أي حسابية ، منطقية وأبجعددية . يمكنها أن تكون مركزية بداخل ماكرو - إجراء والكود - المفتوح⁽²⁾ (Open-code) أو شاملة (كلية) في جميع ماكرو - الاجراءات وفي الكود المفتوح . يجب أن يصرّح عن جميع متحولات التأويل ، المركزية

(1) مُصطلح معروف في 2.6

(2) الكود المفتوح : (Open code) قسم من كود المصدر يكون موجوداً خارج وبعد الماكرو - تعريفات

والكلية ، قبل إستعمالها . ويتم التصريح حسب نوع المتحولة A ، B أو C :

LCLA ... LCLB ... LCLC ... (مركزية)
GBLA ... GBLB ... GBLC ... (كلية)

لا يجب أن يُصرَّح عن متغيرات الماكرو بتعريف (فقرة 1.2.22) . عند التصريح
توضع المتحولات A و B في صفر ويجري إعداد المتحولة من النوع C في « سلسلة فارغة
من السبات » .

لا يمكن بلوغ متحولة مُصرَّح عنها على أنها مركزية إلا في نفس الماكرو تعريف وفي
الكود المفتوح . أما المتحولة المصرَّح عنها « شاملة » (كلية) فيمكن بلوغها من ماکرو
تعريفات أخرى .

يمكن أن تكون متحولة التأويل المشروط عبارة من متحولة مؤشرة ، وفي هذه
الحالة يجب أن يتم التصريح عنها في مستوى LCLx أو GBLx ، كما نُصرَّح عن الجدول
في فورتران . هكذا فإن :

LCLA &TAB(20)

تصرَّح عن TAB & كجدول من 20 عنصراً نستطيع بلوغه بواسطة أحد
الأشكال التالية :

&TAB (تعبير حسابي) مثلاً : &TAB(&I+3)
&TAB (متحولة مؤشرة) مثلاً : &TAB(&VAR(&I))

التعبير الذي يعطي قيمة المؤشر يجب أن يكون إيجابياً وأن لا يزيد عن حجم
الجدول المشار إليه في التصريح .

الثوابت الحسابية عبارة عن أعداد صحيحة بإشارة أو بدون إشارة حيث يجب أن
تكون قيمتها بين : ($2^{31} - 1$ و 2^{31}) .

تأخذ الثوابت المنطقية القيمة 0 أو 1 التي تناسب الغلط والصحيح . الثوابت من
النوع سلسلة سمات تحتوي على عدد من 0 إلى 255 سمة محصورة بداخل فواصل عليا ،
ويمكن أن تكون مؤشرة .
أمثلة :

(4) 'ABCDEF' ، تعادل 'D' ،

(2,3) 'ABCDEF' تعادل 'BCD' ،

المؤشر الأول يعطي الموقع الأولي للسلسلة الثانوية والثاني يعطي طولها .

2.1.22 . أسماء الأوسمة

منطقة الرمز من أمر تأويل مشروط يمكن أن تحتوي على وسم تأويل مشروط . لأنه عبارة عن رمز يبدأ بالنقطة « . » ويسمح ببلوغ أمر تأويل مشروط . لأسماء الوسم مدى مركزي .

3.1.22 . أوامر التخصيص SETx

تقوم بتخصيص قيمة معينة إلى متحولة التأويل المشروط ، تتعلق بنوع المتحولات A ، B ، C ، وتتم بواسطة SETA ، SETB أو SETC . نشير إلى أن متحولة التأويل التي تحصل على التخصيص موجودة في المنطقة المحجوزة عادة للوسم . ولو افترضنا أن &A ، &B و &C هي متحولات من النوع A ، B و C . نكتب :

منطقة المعامل	منطقة العملية	منطقة الرمز
تعبير حسابي	SETA	&A
(تعبير منطقي)	SETB	&B
'تعبير أبجدي'	SETC	&C

وبشكل عام ، بحسب التعبير ونُحْزَن القيمة الناتجة في متحولة التأويل الموجودة لجهة اليسار .

التعابير الحسابية

وتُكتب بواسطة المؤثرات + ، - ، * و / (قسمة صحيحة بدون باق) . التقييم يتم من اليسار إلى اليمين بقواعد الأولوية العادية . أمثلة :

القيمة التي نأخذها من المتحولة

&A1	SETA	10	10
&A1	SETA	&A1+1	11

التعابير المنطقية

تُكتب بداخل أهلة بواسطة المؤثرات NOT ، AND و OR المذكورة في الترتيب التناقصي للأولويات . ويفضل وجود مؤثرات العلاقة بإمكاننا إجراء المقارنات بين التعابير الحسابية .

GT	GE	NE	EQ	LE	LT	مؤثرات علاقة :
>	≥	≠	=	≤	<	المعنى

يجب أن تكون المؤثرات محاطة بفراغات .

أمثلة :

```
&B4 SETB (&B1 OR &B2 AND &B3)
&B5 SETB (&A1 GT &A2)
&B6 SETB ('&C' EQ 'ALLOC')
```

تعايير من نوع سلسلة سمات
هي عبارة عن مجموعات من الثوابت والمتحولات من النوع الابعدي المحصورة
بداخل فواصل عليا . المؤثر « . » (نقطة) يسمح بإجراء عمليات الاتحاد⁽¹⁾ . الترميز
المؤشر يسمح باستخراج السلاسل الثانوية .
أمثلة :

			القيمة التي تأخذها التمييز
&C1	SETC	'CHA'	CHA
&C2	SETC	'&C1'	CHA
&C3	SETC	'&C1'. 'INE'	CHaine
		ou '&C1.INE'	
&C4	SETC	'CHaine' (2,5) الطول ↑ الرتبة ↓	HAINE
&C5	SETC	'&C4' (1,3). '&C4' (5,1)	HAIE
&C6	SETC	'L' 'NOM'	L'NOM
&C7	SETC	'5'	5 (caractère)
&C8	SETC	'&C7..25'	5.25 (un seul point)
&C9	SETC	'&A+10'	si &A = 10 alors
		ou '&A..+10'	10+10 et non 20
&C10	SETC	'&C1&C1'	CHACHA
		ou '&C1.&C1'	

نشير (&C10) إلى أن النقطة في عملية الاتحاد هي إختيارية عندما نجمع بين متحولتين
من السمات لأن الفاصل & لا يسمح بقيام أي نوع من الإبهام .
عندما تدخل المتحولات من النوع A إلى يمين الأمر (&C9) SETC ، فإن قيمة
المتحولات تستبدل بالمتحولات ولكن بدون إجراء لأية عملية .
التعايير من النوع سلاسل السمات هي مهمة لأنها تسمح بإنشاء رموز أو بناء
تعليمات إتحاد متتالية . هناك أمثلة توضح إستعمالها عند دراسة الماكرو - إجراءات .

(1) عملية الربط - جمع سلسلتين ABCD و EF معناه تشكيل السلسلة ABCDEF .

4.1.22 . أوامر التفرع إلى أوسمة التأويل
التفرع الإلزامي يتم بواسطة AGO والتفرع المشروط بواسطة AIF . ويكتبان :

وسم للتأويل المشروط AGO [وسم التأويل المشروط]
وسم تأويل مشروط (تعبير منطقي) AIF [وسم تأويل مشروط]

أمثلة :

AGO .SUITE SUITE إلى
AIF ('&C' EQ 'OUI').ET1 إذا &C تعادل OUI (نعم)
إذهب إلى ET1 ، وإلا تابع بالتالي .

5.1.22 . الأمر ANOP
هو أمر « بدون عملية » يسمح بتعريف وسم معين (Label) . ويُستعمل بشكل خاص عندما نرغب بإجراء تفرع إلى أمر (توجيه) SETx ، ويكون حقل الوسم العادي مشغولاً بمتحولة .

	AGO	.SUITE
	---	-----
.SUITE	ANOP	
&VAR	SETA	&VAR+1

6.1.22 . أمثلة على إستعمال التأويل المشروط
سنذكر عدة أمثلة عند دراسة ماكرو - الإجراءات . هنا نكتفي بتفصيل بعض النقاط

مثل 1

نرغب ، خلال تنفيذ البرنامج ، بإجراء تأويل مجموعة من التعليقات (طباعة وسيطية مثلاً) بإلغاء تعليقات التأويل النهائية دون سحب البطاقات المناسبة لها . سنخضع إذاً تأويل هذه التعليقات للقيمة التي تأخذها متحولة التأويل التي تدعى هنا & TEST

&TEST	SETA	1	(مرحلة البدء بالعمل)
	----	----	
	AIF	(&TEST EQ 0).SAUT	
	---	---	تعليقات للتأويل
	---	---	خلال مدة الاختبار
.SAUT	---	---	

بجعل المتحولة &TEST تعادل صفرًا نكون قد ألغينا تأويل هذه التعليقات .

مثل 2

إنشاء نصّ معيّن .

التأويل المشروط يمكن أن يُستعمل لإنشاء نصّ متحوّل من تأويل إلى آخر . يمكن لهذا النص أن يكون رمزاً أو تعليمة .

يؤدي إلى توليد الأمر :
 &NO EQU &NO
 R1 EQU 1
 إذا كانت المتحولة &NO تعادل 1

2.22 . الماكرو - إجراءات

باستعمال الماكرو إجراءات نجد أولية التأويل المشروط فائدتها :

الماكرو إجراء هو عبارة عن برنامج يحمل اسماً مؤلفاً من سلسلة من التعليقات وأوامر التأويل المشروطة وغير المخصورة بالأوامر MACRO و MEND .
 مثلاً : الماكرو تعريف التالي :

لائحة المتغيرات الشكلية اسم
 MACRO
 SOMME &U,&V,&W
 { L 1,&U
 A 1,&V
 ST 1,&W
 MEND
 سطر نمطي
 جسم
 الإجراء

سيكون الماكرو تعريف موجوداً خارج البرنامج (open code) الذي يُراجع .
 بإمكان الماكرو تعريف أن يكون موجوداً في مكتبة المُستعمل أو مكتبة المؤول .
 الماكرو تعليقات هي إذاً السطر من البرنامج الذي يطلب من المؤول إدخال نص النموذج في البرنامج باستبدال المتغيرات الشكلية بالمتغيرات الفعلية .
 مثلاً :

يُولد المتالية
 SOMME A,B,C
 L 1,A
 A 1,B
 ST 1,C
 المتغيرات الوسيطة الفعلية

نفترض عندئذٍ بأن هذا النظام ، المزود بالتأويل المشروط ، يسمح بإنشاء نماذج ستاندارد لبرامج يقوم المؤول بجعلها متوافقة مع كل حالة خاصة حسب قيم تحولات التأويل المشروط .

1.2.22 . تنقل المتغيرات

كما في حالة البرامج الثانوية ، المتغيرات الشكلية هي متغيرات السطر النموذجي في الماكرو تعريف والمتغيرات الفعلية هي متغيرات الماكرو تعليمية . المتغيرات الشكلية هي رموز تسبقها السمة «&» .

يتكوّن السطر النموذجي في الماكرو تعريف على الشكل التالي :

اسم الإجراء	لائحة المتغيرات الشكلية
PROC	&U,&NO=3,&QTE=,&V,&RES=5,&W,&X

قيم نحو النقصان (0 أو سلسلة فارغة إن لم يجرّ تحديدها) .

المتغيرات الشكلية هي على نوعين

- متغيرات الوضع : &X و &U, &V, &W في المثل ،
- متغيرات الكلمة المفتاح : &NO ، &QTE و &RES . وغيّرها يكون أسائها متبوعة بالرمز « = » وربّما بالقيمة التي تأخذها نحو النقصان ، قيمة تساوي « السلسلة الفارغة » في حال عدم تحديدها . ويتكوّن سطر نداء الماكرو تعريف كما يلي :

اسم الإجراء	لائحة المتغيرات الفعلية
PROC.	RES = 6, A, B, QTE = 4,,D

1- متغيرات مرتبطة بالمتغيرات الشكلية - من حيث مواقعها في اللائحة . لدينا هكذا التناسب بين A و &U ، B و &V ، D و &X . إن فاصلتين متتاليتين تشيران إلى غياب متغير الوضع .

2- متغيرات الكلمة المفتاح : الوصل بين المتغيرات الشكلية والفعلية القائم بفضل تشابه الاسم . هذه العناصر يجب أن يليها الرمز « = » وربّما قيمة تعدّل القيمة المحددة نحو النقصان . في مثلنا تأخذ RES القيمة 6 ، QTE القيمة 4 وتحفظ NO بالقيمة 3 نحو النقصان .

3- قد تكون لوائح متغيرات محاطة بأهلة . لتأخذ الماكرو تعليمية :

PROC.1 (A, B, C, D), K = (E, F, G, H)

والسطر النموذجي المناسب :

PROC 1 &POS,&K=

تتكون المتغيرات الفعلية بواسطة اللاتحين (A, B, C, D) و (E, F, G, H). أما &POS(3) فيُستبدل عندئذٍ بـ C خلال انتشار الماكرو تعليمة. كذلك يُستبدل &K(2) بـ F. بإمكان لوائح المتغيرات أن تكون ذات أطوال متغيرة، وسنرى أن الخاصية &POS N' تسمح بمعرفة طول اللائحة المرتبطة بـ POS.

2.2.22 . تطبيق

المثل التالي يقوم بتوليد تعليمات تسمح بجمع n خلية من الذاكرة منقولة إلى ماكرو الإجراء بواسطة لائحة &RES. ستحتوي على النتيجة و &NB تمثل عدد العناصر المطلوب جمعها. المؤشر المركزي &I يُستعمل لمراجعة مختلف عناصر اللائحة.

```

1      MACRO
2      SOMME  &MEM,&RES,&NB=&, &REG=
3      LCLA  &I
4      L      &REG,&MEM(1)
5      &I   SETA  I
6      .BOUCLE ANOP
7      &I   SETA  &I+1
8      AIF  (&I GT &NB).FIN
9      A    &REG,&MEM(&I)
10     AGO  .BOUCLE
11     .FIN ST    &REG,&RES
12     MEND

```

```

000060 5830 C074 00074 64      SOMME (A,B,C,D),X,NB=4,REG=3
000064 5A30 C078 00078 65+    L      3,A
000068 5A30 C07C 0007C 66+    A      3,B
00006C 5A30 C080 0C080 67+    A      3,C
000070 5030 C084 00084 68+    A      3,D
                                69+    ST    3,X

```

```

000074      72 A      DS    F
000078      73 B      DS    F
00007C      74 C      DS    F
000080      75 D      DS    F
000084      76 X      DS    F

```

3.2.22 . الأمر MEXIT

يسمح بوقف تأويل الماكرو تعريف. من الممكن إعتباره معادلاً للتفرع إلى الأمر

. MEND

4.2.22 . الأمر ACTR

يسمح بمراقبة عدد AIF وAGO الجاري خلال التأويل المشروط. ويكتب:

(تعبير حسابي) ACTR

يؤدي إلى توليد عداد يعادل مضمونه قيمة التعبير الحسابي. يمكن أن يكون العداد مركزياً للماكرو تعريف أو شاملاً. في كل مرة يجري فيها تنفيذ AIF أو AGO بواسطة المؤول، فإن العداد المناسب لهذا القسم من البرنامج يُخفّض واحداً من مضمونه. وعندما يبلغ الصفر، فإن المؤول يخرج من الماكرو تعريف (فعل معادل لـ

MEXIT (أو يُوقف التأويل إذا كان ذلك متعلقاً بعدد شامل . هذا الأمر يسمح بتحديد عدد الحلقات التي تجري في مرحلة ما قبل التأويل .

5.2.22 . الأمر MNOTE

يمكن أن يُستعمل من قبل المبرمج لتوليد رسالة الخطأ الخاصة به أو طباعة قيم وسيطية مأخوذة من متحولات التأويل .

ويمكن أن يُكتب بعدة أشكال : وسم تأويل

- | | | | |
|-----|---------------------------|-------|-----------------|
| (1) | étiquette
d'assemblage | MNOTE | code, 'message' |
| (2) | étiq. assem. | MNOTE | , 'message' |
| (3) | étiq. assem. | MNOTE | *, 'message' |
| (4) | étiq. assem. | MNOTE | 'message' |

الكود هو عبارة عن تعبير حسابي بقيمة محصورة بين 0 و 255 يربط مستوى من الخطأ بالرسالة . في الشكل 2 يُفترض بالكود أن يكون مُعادلاً لـ 1 . لا تُطبع الرسالة من ضمن رسائل الخطأ إلا إذا كان الكود الذي يشير إلى درجة الحقيقة هو أعلى من أو يعادل الكود المعتمد من المؤلف .

الشكلان 3 و 4 يولدان الرسالة كمجرد ملاحظة . .

6.2.22 . الملاحظات :

من الممكن إدخال ملاحظات في ماكرو التعريفات على الشكل التالي :

- * COMMENTAIRE GENERE
- * COMMENTAIRE NON GENERE

7.2.22 . الدوال من النوع الذاتي (Intrinsic)

&SYSLIST

تسمح ، داخل الماكرو تعريف ، بتسمية متغيرات الموقع الموجودة داخل ماكرو تعليمة النداء . وتُكتب بمؤشر أو بمؤشرين يمكن أن يكونا عبارة عن تعابير حسابية من نوع ذلك الذي رأيناه في الفقرة 3.1.22 . سنختبر إستعمالها بالخاصية N° .

&SYSLIST(&I) تشير إلى المتغير الفعلي الخاص بالموقع رقم i من التعليمة . يمكن أن يكون هذا المتغير الفعلي عبارة عن لائحة (حسب الفقرة 3.1.22) . في هذه الحالة ، سنسمي العنصر رقم z من اللائحة بالرتبة &I بواسطة &SYSLIST(&I,&J) . في المثل المذكور في الفقرة 2.2.22 &SYSLIST(1,2) تعني المتغير B ، و &SYSLIST(2) تعني X .

&SYSLIST(0) تعني الوسم الموجود قبل الماكرو تعليمة الخاصة بالنداء . هذه المهمة تسمح بتفادي تسمية المتغيرات .

&SYSNDX

هي عبارة عن عداد من أربعة أرقام عشرية ، وهو مركزي ضمن ماكرو - تعريف ، وتزداد قيمته عند كل استعمال جديد للماكرو . لا يمكن أن يُستعمل وحيداً ولكن يُمكن أن يتّحد مع رمز ما . هذه هي الوسيلة لتوليد أوسمة مختلفة عند كل نداء للماكرو - التعريف وتسمح بتفادي الأخطاء في التأويل والناجئة عن تعريف الرموز .
مثلاً :

لنفترض الماكرو - تعريف التالي :

```
MACRO
PROC    &A,...
&A&SYSNDX  ----
        ----
R&SYSNDX   ----
        ----
MEND
```

النداء الأول يتم بواسطة PROC ETIQ,...

ETIQ0001 تأخذ القيمة المتحولة &A&SYSNDX

R0001 تأخذ القيمة المتحولة R&SYSNDX

في النداء الثاني بواسطة PROC ETIQ,...

ETIQ0002 تأخذ القيمة المتحولة &A&SYSNDX

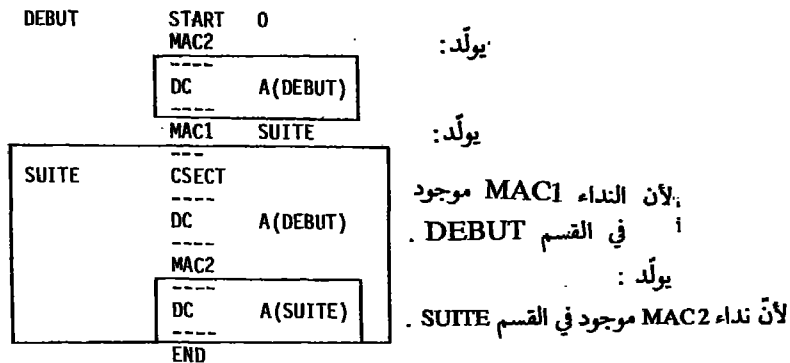
R0002 تأخذ القيمة المتحولة R&SYSNDX

&SYSECT

تسمح بتعريف اسم القسم حيث توجد الماكرو - تعليمة المُنادية . المثل التالي يوضح

ذلك :

```
MACRO
MAC1    &ETIQ
&ETIQ   ----
        CSECT
        DC    A(&SYSECT)
        ----
MAC2
        ----
MEND
MACRO
MAC2
        ----
        DC    A(&SYSECT)
        ----
MEND
```



&SYSPARM

يعطي وسيلة الرجوع إلى المتغير SYSPARM لبطاقة //EXEC في Job JCL (Control Language : لغة مراقبة العمل) .
مثلاً .

```
//      EXEC ASMC,PARM=SYSPARM(DEBUG)
//ASM.SYSIN DD *
TEST   START 0
-----
AIF     ('&SYSPARM' NE 'DEBUG'). (قفزة)
----- ولادة تعليقات
----- تنفيذ وتقرير
.SAUT   ANOP
```

&SYSTIME

يعطي ساعة التأويل بواسطة خمس ساعات : h.h.mm

&SYSDATE

يعطي التاريخ بواسطة ثمان ساعات : mm/jj/aa

8.2.22 . الخاصيات

مفهوم الخاصية المرتبط بمعطى أو بتعليمة جرت إثارته في الفقرة 2.3.6 .. كما

إستعملنا الخاصية - طول (فقرة 3.2.7) . يسمح المؤول لنا باستعمال خاصيات أخرى حيث البعض منها يجد إستعمالاً بسبب وجود إمكانيات التأويل المشروط .

الخاصية : TYPE T'

وقيمتها سمة أبجدية حسب نوع الرمز المطبقة عليه . إذا كانت NUM ، مثلاً ، عبارة عن ثابتة عشرية موسعة ، فإن قيمة T'NUM ستكون Z . الحرف الذي يُميز النوع هو نفسه المُستعمل في الأوامر A:DC تناسب ثابتة عنوان من نوع A ، بينما B تناسب ثابتة منطقية . . . ونضيف التناسبات التالية :

G	ثابتة بفاصلة ثابتة وطول محدد ظاهر
K	ثابتة بفاصلة متحركة وطول محدد ظاهر
R	ثابتة عنوان بطول محدد ظاهر
I	تعليلة - آلية
M	ماكرو تعليلة
W	CCW
J	اسم قسم
T	رمز خارجي
N	قيمة تعريف أوتوماتيكي
O	سمة مخلوطة
	{ تتعلّقان بمتغيرات الماكرو تعليلة

الخاصية LONGUEUR L' (طول)

جرت دراستها في الفقرة 3.2.7 .

الخاصية مقياس S'

عبارة عن قيمة رقمية تتعلّق بنوع الرمز .

- لعدد عشري (نوع P أو Z)

عبارة عن عدد الأرقام في القسم الكسري .

- لعدد بفاصلة متحركة (أنواع L, E, D أو K)

إنه عدد الأصفار السادس عشرية في يسار القسم العشري (الوزن الأكبر) .

- لعدد بفاصلة ثابتة (الأنواع M ، F أو G)

عبارة عن القوة 2 التي يتم ضرب قيمة الثابتة بها . وتشير إلى عدد البتات في

القسم الكسري إذا كان إيجابياً ، وعدد البتات المتروكة إذا كان سلبياً .

الخاصية قسم صحيح P
عبارة عن عدد يتعلّق بـ S' و L' .

- لعدد عشري من نوع P $P = 2 * L' - S' - 1$
- لعدد عشري من نوع Z $P = L' - S'$
- لعدد بفاصلة متحركة من نوع K , L , E , D $L' \leq 8$ مع $I' = 2 * (L' - 1) - S$
- لعدد بفاصلة متحركة من نوع L $L' > 8$ مع $P = 2 * (L' - 1) - S' - 2$
- لعدد بفاصلة ثابتة من نوع G , F , H $I' = 8 * L' - S' - 1$

الخاصية عدد السمات K'
وتُطبّق فقط على مُتغيّرات الماكرو - تعلّمة وأيضاً ، بإشراف OS ، على الرموز المتحوّلة .. & وعلى الدوال الذاتية (من نوع intrinsic) . وتعطي عدد سمات الرمز التي تطبّق عليه .

أمثلة : في مثل الفقرة 2.2.22 : $K' \& MEM = 9$

```
&A SETA 253 : K'&A = 3,
&B SETB 0 : K'&B = 1,
&C SETC 'ALPHA' : K'&C = 5.
```

الخاصية عدد العناصر من اللائحة N'
وتنطبق فقط على متغيّرات الماكرو - تعلّمة ، وتعطي عدد عناصر اللائحة .
مثلاً :

```
PROC &A,&B,&K=      خط نموذج
PROC (1,2,,4),U,K=3  ماكرو تعلّمة
N'&A = 4             (يتم تعداد السمات غير الموجودة)
N'&SYSLIST = 2        متغيّرات الموقع
N'&SYSLIST(1) = 4.
```

9.2.22 . أمثلة عن الماكرو - تعريفات
الماكرو - تعريف التالي يسمح بتوليد الأوامر (التوجيهات) المُعادلة لـ REQUI

12	MACRO	
13	EQU REG	
14 *	MACRO D*	EQUIVALENCE REGISTRES ماكرو معادل المرافف
15	GBLA	&NO
16	SETA	1
17	ATF	(&NO GT 15).FIN
18	EQU	&NO
19	SETA	&NO+1
20	AGO	*SI
21	.FIN	MEND

وُئُولد الكود التالي :

	33	EQU REG	
	34+*	MACRO	D
00001	35+R1	EQU	1
00002	36+R2	EQU	2
00003	37+R3	EQU	3
00004	38+R4	EQU	4
00005	39+R5	EQU	5
00006	40+R6	EQU	6
00007	41+R7	EQU	7
00008	42+R8	EQU	8
00009	43+R9	EQU	9
0000A	44+R10	EQU	10
0000B	45+R11	EQU	11
0000C	46+R12	EQU	12
0000D	47+R13	EQU	13
0000E	48+R14	EQU	14
0000F	49+R15	EQU	15

الماكرو- تعريف PROLOGUE يسمح بشحن واحد أو عدة مرادف قاعدة مخزنًا مفهوم البرنامج المتادي حسب المعايير العادية المحددة في الفصل 21 . وهو يُعرّف في نفس الوقت منطقة SAVE AREA للبرنامج الجاري . عنوان القاعدة الذي جرى اختياره هو عنوان نقطة الدخول إلى البرنامج . ويرد الكود المُولد على الصفحة التالية .

```

14      MACRO
15      PROLOGUE  &ABASE=,&RBASE=
16      EQU      *
17      LCLA     &I,&D
18      LCLC     &CH
19      STM      14,12,12(13)
20      *        CONSTRUCTION DE USING إنشاء مصنع
21      &CH      SETC  '&RBASE(1) '
22      &I       SETA  2
23      *T1      AIF   (&I GT N*&RBASE).SUIT1
24      &CH      SETC  '&CH'.',',',*&RBASE(&I)'
25      &I       SETA  &I+1
26      AGO      *T1
27      *SUIT1   ANOP
28      USING    &ABASE,&CH
29      LR       &RBASE(1),15
30      ST       13,SAVEAREA+4
31      LR       2,13
32      LA       13,SAVEAREA
33      ST       13,8(2)
34      *        CONSTRUCTION DES CHARGEMENTS DES REGISTRES DE BASE
35      &I       SETA  2
36      *T2      AIF   (&I GT N*&RBASE).SUIT2 إنشاء شحن المرادف القاعدي
37      L        0,=F'4096'
38      *T3      AIF   (&I GT N*&RBASE).SUIT2
39      AR       15,0
40      LR       &RBASE(&I),15
41      &I       SETA  &I+1
42      AGO      *T3
43      *SUIT2   ANOP
44      B        **+76
45      SAVEAREA DS 10F
46      MEND

```

من المفيد دراسة أمثلة الماكرو تعريفات المذكورة في كتاب إ. تابوريه Y.Tabourier ، أ. روشفلد Rochfeld ونس. فرانك C. Frank . إنها عبارة عن ماكرو تعريفات تسمح ببناء برنامج مؤول بصورة بنوية مركبة . والكتاب يعرض للماكرو WHILE زائد شرط ، DO ، ENDWHILE ، IF ، THEN ، ELSE ، BLOCK ، ENDBLOCK بالتفصيل ، وتقوم هي باستدعاء 2 ماكرو تديران مكدياً من المؤثرات .

```

000000 90EC D00C
000004 18CF C024
000008 50D0 C020
00000A 182D C020
00000C 41D0 C008
000010 50D2 C090
000014 5A02 C090
000018 1A50 C068
00001A 1A5F C068
00001C 47F0 C068
000020

52+PROLOG      00030
53+PROLOG      0000C
54+             00030
55+             00024
56+             00320
57+             00008
58+             00090
59+             00068
60+             00068
61+             00068
62+             00068
63+             00068
64+             00068
65+SAVEAREA DS

PROLOGUE ABASE=P1,RB,SE=(12,11)
EQU #
STM 14,12,12(13)
USING P1,12,11
LR 12,15
ST 13,SAVEAREA+4
LR 2,13
LA 13,SAVEAREA
ST 13,8(2)
O=F,4096,
LR 15,0
LR 11,15
LR 11,76
B *+76
18F

```

```

C0000C 90EC D00C
C00004 18CF C01C
C00008 50D0 C01C
C0000A 182D C018
C0000C 41D0 C018
C00010 50D2 C008
C00014 47F0 C060
C00018

52+PROLOG      00030
53+PROLOG      0000C
54+             00030
55+             0001C
56+             0001C
57+             00018
58+             00018
59+             00008
60+             00060
61+             00060
62+SAVEAREA DS

PROLOGUE ABASE=P1,RBASE=12
EQU #
STM 14,12,12(13)
USING P1,12
LR 12,15
ST 13,SAVEAREA+4
LR 2,13
LA 13,SAVEAREA
ST 13,8(2)
B *+76
18F

```

23 . نصائح في البرمجة

ليس هدفنا عرض طريقة في البرمجة تشبه البرمجة الإنشائية ، ولكن ببساطة إعطاء بعض النصائح الناتجة عن الخبرة العملية لمختلف الطرق . هذه الملاحظات يمكن أن توسع لتشمل جميع أنواع المؤول وفي بعض الأحيان تنطبق على اللغات المتطورة .

1.23 . تركيبة المعالجة

1.1.23 . البرمجة الزجلية

هي عبارة عن قاعدة عامة في البرمجة . هناك فائدة من تقسيم المسألة إلى زجل (أقسام) صغيرة قدر الإمكان . كل زجلة تحل مهمة معينة والبرنامج الرئيسي يؤمن ترابط الأقسام فيما بينها . ولقد عرضنا في الفصلين 20 و 21 . طريقة استعمال وسائل التقطيع وإنشاء البرامج - الثانوية .

2.1.23 . تقديم وإعداد

البرنامج بلغة المؤول هو عادة عبارة عن نص غير واضح ، ويجد المصمم صعوبة في تعديل وإعادة قراءة ما كتبه منذ اللحظة التي يترك فيها برنامجه جانباً لبعض الوقت . يجب إذا كتابة الملاحظات بعد كل تعليمة لتوضيح نص البرنامج . الأوامر SPACE N (إدخال عدة أسطر n بيضاء) ، EJECT (عبور إلى الصفحة التالية) و PRINT NOGEN (إلغاء توليد كود الماكرو تعليقات) تسمح بتسهيل نص البرنامج بجعله أكثر وضوحاً .

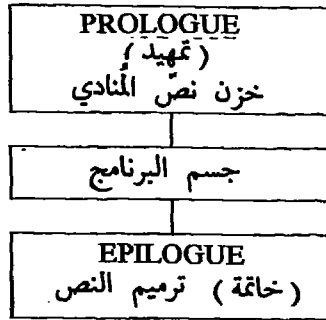
البرنامج المؤود بملاحظات يبدأ بتحديد مهمة الزجلة ، وروابطها مع الزجل الباقية كما يحتوي على أسماء ومهمة المتحولات والمراسف المستعملة .

2.23 . تركيبة الزجلة

1.2.23 . التمهيد والخاتمة (Prologue and epilogue)

بإمكاننا اعتبار كل برنامج وكأنه برنامج ثانوي لبرنامج آخر . الزجلة الرئيسية هي عبارة عن برنامج ثانوي من نظام التشغيل ويجب عليها أن تخزن نتائج البرنامج النهائي .

تقنية الخزن وترميم نصّ المُنادي هي أساسية وقد جرى تعريفها في الفصل 21 . بإمكاننا إنشاء كل زجلة على الشكل التالي :



هناك فائدة للمبرمج في تحقيق التمهيدات والخاتمة الخاصة به حسب القواعد المتفق عليها والمذكورة في الفصل 21 . الاتصال بين الزجل المكتوبة في اللغات المختلفة سيكون مبسّطاً وأكثر من خطاً سيتم تفاديه باستعمال مناسب للمراصف . لقد ذكرنا مثلاً في الفصل 22 الماكرو - تعليمة PROLOGUE التي تحلّ هذه المسألة وتوفّر على المبرمج كتابة صعبة للتعليقات الأولية .

2.2.23 . جسم البرنامج

يتألف من تعليقات قابلة للتنفيذ ومن معطيات . سنضع المعطيات بعد التعليقات . استعمال الأمر LTORG سيسمح لنا بوضع تأويل الثوابت من نوع حرفي في المكان الذي نرغب فيه . المتحولات والثوابت ستكون إذاً متراصة ، مما يجعلها متجاورة في كل dump وستسمح بإجراء تقسيم سهل إلى أقسام إذا كنا نرغب بجعل البرنامج مُبسّطاً للتعديل والاختبار . سنستعمل عند الحاجة أوامر حجز مكان من الذاكرة بواسطة DC أكثر من بواسطة DS معيّنين بهذه الطريقة منطقة من الذاكرة بقيمة سوف يمكننا مراقبتها في dump (دلق) .

إستعمال المرجعيات الرمزية

إنّ كتابة LR1,2 تعود عملياً إلى العمل بلغة الآلة . وفي المقابل فإنّ كتابة LR1,R2 بعد تعريف الرمزين R1 و R2 بواسطة EQU معناها إستعمال إمكانيات ومرونة الترميز ، والمرجعان R1 و R2 يظهران في جدول الرموز . من الأفضل أيضاً إعطاء المراصف والمتحولات أسماء مكوّدة حرفياً كما جرى في أمثلة الفصل 15 . فليس من المزعج أكثر من قراءة التعليقات التي تذكر المراصف بشكل ظاهر .

هكذا ، فكتابة $B * + 14$ تؤدي إلى سيئة تكمن في تجميد البرنامج ، ويصبح من

غير الممكن إدخال تعليقات جديدة بين العناوين * و14+ * دون تعديل تعليقات
التفريع . لذا فمن الأفضل تعريف وسم ALPHA وكتابة B ALPHA . الكتابات من
النوع n + * لا يجب أن تستعمل إلا داخل الماكرو - تعريفات . وختاماً يجب على
البرنامج أن يكون دائماً مكتوباً مع أخذ التعديلات اللاحقة بعين الاعتبار إضافة إلى
مسائل الصيانة ..

هكذا يجب تعريف جميع العناصر القابلة للتعديل في البرنامج بواسطة EQU .
هذا الأمر هو شديد الأهمية . وفي حالة التعديل فهو يسمح بتخفيض عدد التغيرات
المطلوب إجراؤها . ويقدم فائدة تكمن في جعل التعليقات « مزودة بملاحظات » . إن
التمرين 8.13 يوضح لنا ذلك .

الخاصية - طول

تسمح بجعل البرنامج يحتوي على متغيرات . كل تعديل على طول المنطقة لن
يؤثر على التعليقات التي تذكر هذا الطول بواسطة L'ZONE .

تركيبة منطقة المعطيات

بدلاً من مراجعة أقسام (field) نفس المنطقة بواسطة المسافة بالنسبة لبداية
المنطقة ، من الأفضل تخصيص (بواسطة EQU) أسماء رمزية لمختلف هذه الأقسام .
كل تعديل على التركيبة يصبح عندئذ سهلاً . يُوضح لنا التمرين 2.8 تعريف تركيبة
كهذه .

إستعمال الكود الحرفي

يترك للمؤول مهمة تعريف الثوابت الضرورية دون إسهاب . هذه الثوابت يمكن
أن تكون مجموعة في المكان المطلوب بواسطة الأمر LTORG .

كتابة الأوسمة

سنعرّف الأوسمة بواسطة الأمر DSOH . نتأكد من تسطير (إصطفااف) حدود
نصف - كلمة والوسم لن يعود مرتبطاً بالتعليمة الموجودة في الجهة المقابلة له . سيصبح
ممكناً عكس بعض التعليقات بواسطة معالجة بسيطة للبطاقات .

إستعمال المراسف

قبل أية عملية برمجة يجب التنقيب عن الخيارات التي يقوم بها النظام لاستعمال
المراسف . وقد جرى عرض ذلك في الفصل 21 . وللمبرمج فائدة في إجراء نفس
الاختيار لأسباب تتعلق بالتوافق . فلنذكر أن OS :

يشحن في R15 عنوان نقطة الدخول ،

في R14 عنوان العودة ،

في R13 عنوان المنطقة SAVE AREA .

وستعمل R0 لارسال نتيجة مهمة من نوع (FUNCTION في فورتران) ،

و R1 لإرسال عنوان لائحة متغيرات إلى برنامج ثانوي .

بعض التعليقات (TRT, EMDK) تستعمل المرصفين R1 و R2 . سيختار المبرمج مرادف القاعدة من ضمن المرادف 12 ، 11 و... ومرادف العمل من ضمن المرادف 3 ، 4 ، ...

إستعمال الماكرو - لغة (MACRO-language)

باستعمال الماكرو لغة فإن المؤول يقترب من اللغة المتطورة . وهي تسمح للمبرمج بأن يكون مزوداً بوسائل إعداد البرنامج وجعله إنشائياً (مركباً) . وسيكون بإمكانه ، مثلاً ، إنشاء ماكرو - تعريف يسمح له بمتابعة أثر البرنامج عند التنفيذ بواسطة طباعة الأوسمة خلال مرحلة الاختبار . عند التأويل النهائي فإن توليد الماكرو - تعليمة سيتم إلغاؤه بواسطة تعديل بسيط لقيمة متحولة التأويل . ولن تولد أوسمة بواسطة ETIQ DS OH . بإمكان المبرمج أن يقوم أيضاً بإنشاء ماكروتعريفات تولد مثلاً تعليمات من نوع WHILE ، DO ، ENDDO ... وبإمكان الأوسمة أن تخفي من النص المطلوب تأويله ويصبح البرنامج أكثر إنشائياً .

وفي النهاية فإن الزجلة يمكن أن تحصل على التركيبة التالية :

MACRO-DEFINITIONS
COMMENTAIRES
EQU ...
PROLOGUE
CORPS
EPILOGUE
ZONE DE DONNÉES

ماكرو تعليمات
ملاحظات
EQU...
مقدمة
خاتمة
جسم البرنامج
منطقة المعطيات

3.23 . الخلاصة

بشكل عام لا نؤيد المبالغة في استعمال الحيل والحلق من قبل المبرمج . فالبرنامج « المتحليل » هو غامض على العموم بالنسبة للقارئ المبتدىء ، وأحياناً تقترب الحيل من الإضمار المبهم ويمكننا هنا تصور المشاكل التي قد تعترض عمل فريق صيانة البرامج .

في لغة المؤول تختلف المسألة نوعاً ما . فبالإمكان إقامة عدد معين من الحيل ضمن نطاق تقنيات الحل وفي هذا الإطار يتعين على المبرمج أن يعرفها . لقد ذكرنا خلال الأمثلة والتباين عدد كبيراً من الوصفات المنتشرة كفاية بشكل يسمح لنا باعتبارها كأدوات أساسية . هذا هو السبب الذي يجعلنا نصرّ علماً ، دراستها من قبل القارئ بعناية واهتمام .

حلول التمارين

النظام 10	النظام 2	النظام 16	تمرين 1.2 -
15	1111	F	
35	10 0011	23	
256	1 0000 0000	100	
1024	100 0000 0000	400	
348.5	1 0101 1100.1	15C.8	

النظام 16	النظام 10	النظام 2	تمرين 2.2 -
3A	58	11 1010	
FFF (=1000-1)	4095	1111 1111 1111	
1A3B	6715	1 1010 0011 1011	
ABC	2748	1010 1011 1100	

تمرين 3.2 - المكمل إلى FFFF : E5C4
المكمل إلى E5C5:2

الطرح بواسطة جمع المكمل إلى 2 (نتحقق ما إذا كان يحق لنا تجاهل
المرحل (النتيجة : 1081 .

1A3B على 32 بته : 3B 1A 00 00
E5C5 على 32 بته : CS E5 FF FF

- تمرين 4.2 - تكويد الإشارة والقيمة المطلقة : $4.16^7 + 16^6 + 15.16^5$ -
 التكويد بالمكمل إلى 2 : $3.16^7 + 14.16^6 + 16^5$ -
 التكويد بالفاصلة المتحركة : $16^1 (15.16^{-1})$ -
 العكس (الضد) بالإشارة والقيمة المطلقة : 41 F0.00 00
 العكس بالمكمل إلى 2 : 3E 10 00 00
 العكس بالفاصلة المتحركة : 41 F0 00 00 (معايير)
 لا يمكن لهذا التمثيل أن يكون تمثيل عدد مكود بالنظام DCB (عشري
 مكود ثنائيًا) .

تمرین 5.2 -

$$\begin{aligned}
 C5\ 03\ 20\ 00 &= -16^5(3.16^{-2}+2.16^{-3}) \\
 &= -\frac{16^5}{16} \cdot 16(3.16^{-2}+2.16^{-3}) \\
 &= -16^4(3.16^{-1}+2.16^{-2}) = C4\ 32\ 00\ 00
 \end{aligned}$$

تمرین 1.8 -

TAB	DC	100AL1(*-TAB+1)
TAB	DC	100A((*-TAB)/4+1)

تمرین 2.8 -

NOSS	DS	OCL13	L'NOSS = 13
SEXE	DS	CL1	L'SEXE = 1
DATE	DS	OCL4	L'DATE = 4
ANNEE	DS	CL2	L'ANNEE = 2
MOIS	DS	CL2	L'MOIS = 2
LIEUNAI	DS	OCL5	L'LIEUNAI = 5
DEPART	DS	CL2	L'DEPART = 2
COM	DS	CL3	L'COM = 3
NO	DS	CL3	L'NO = 3

تمرین 3.8 - تأطير على حد كلمة *

Z1	DS	OF
PRIX	DS	OCL12
QTE	DS	ZL8
	DS	ZL4
	ORG	Z1
Z2	DS	OCL14
NO	DS	F
TEXTE	DS	CL10

تمرین 1.9 -

L'OC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000000				1	CSECT
			000000	2	USING *,12
000000	5810 C92C	0002C		3	L B,D
000004	5833 C02C	0002C		4	L 3,D(3)
000008	0000			5	LR A,D
	*** ERROR ***				
00000A	0000 0300	00004		6	ST D,X'4*(3,C)
	*** ERROR ***				
00000E	5803 0008	00008		7	L A,B*1011*(3)
000012	0000 0000	00040		8	L D,E(B)
	*** ERROR ***				
000016	5801 C040	00040		9	L A,E(B)
00001A	D200 A000 C92C	00000 0002C		10	MVC A(B,C),D
000020	D203 C040 C02C	00040 0002C		11	MVC E(L'D),D
000026	5820 C030	00030		12	L 2,D+L'D
		00000		13 A	EQU 0
		00001		14 B	EQU 1
		0000A		15 C	EQU 10
00002C				16 D	DS 5F
000049				17 E	DS 12F
				18	END

ASSEMBLER DIAGNOSTICS AND STATISTICS

SYM ERROR CODE MESSAGE

5	IF0217	RELOCATABILITY ERROR NEAR OPERAND COLUMN 4
6	IF0217	RELOCATABILITY ERROR NEAR OPERAND COLUMN 2
8	IF0217	RELOCATABILITY ERROR NEAR OPERAND COLUMN 2

NUMBER OF STATEMENTS FLAGGED IN THIS ASSEMBLY = 3
HIGHEST SEVERITY WAS 12

تمرين 1.11 -

حلول أخرى بواسطة «أو المقتصرة» أو «الإزاحات» .

$$\begin{matrix} LA & R,0 \\ SR & R,R \end{matrix}$$

تمرين 2.11 -

LCR R,R

تمرين 3.11 -

نستعمل كون نصف الكلمة موسّعاً إلى كلمة قبل العملية بواسطة انتشار
 بـ 1 ذات وزن قوي .

$$LH \quad R, H'-1'$$

تمرين 4.11 -

$$\begin{matrix} LA & R,2048 \\ LA & R,4095 \\ L & R,=F'4096 \end{matrix} \quad \text{أو :} \quad \begin{matrix} LA & R,4095 \\ LA & R,1(0,R) \end{matrix}$$

تمرين 5.11 -

LA R,4(0,R)

تمرين 6.11 -

نستعمل كون الحركة تتم بايئة بعد بايئة من اليسار إلى اليمين .

$$\begin{matrix} MVI & ZONE,C'x' \\ MVC & ZONE+1(L'ZONE-1),ZONE \end{matrix}$$

تمرين 1.12 -

عدد التكرارات .

$$\begin{matrix} N & DC & \dots \\ R1 & EQU & 3 \\ & & \dots \\ & L & R1,N \\ TRAIT & & \\ & & \dots \\ & & \dots \\ & BCT & R1,TRAIT \end{matrix}$$

 معالجة التكرار .

تمرين 2.12 -

تسمح الماكرو تعليمة SNAP بالحصول على عمليات دلق («dumps») جزئية في الذاكرة . ويجب أن تسبقها ماكرو OPEN (فتح سجل) . في حالتنا الحاضرة يمتد الدلق dump من العنوان SNAPDEB حتى العنوان SNAPFIN . تعطي الكلمة PSW عنوان بداية SNAP . وتعطي الجهة

اليمنى من dump ، حتى يكون ذلك ممكناً ، تفسير محتوى الذاكرة الثنائي على شكل سمات . وسيتمرن القارئ بمحاولة إيجاد محتويات مختلف مناطق البرنامج عبر حساب العناوين من خلال العنوان الأساسي الموجود في المرفص 12 .

(أنظر اللائحة listing في الصفحة اللاحقة) .

تمرين 3.12 -

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000000				1	DEBUT START 0
				2	PRINT NOGEN,DATA
				3	عكس سلسلة من السمات *
		00003		4	WCFK EQU 3
		00004		5	IND1 EQU 4
		00005		6	INC2 EQU 5
000000				7	SNAPDEB DS 2H
000000				8	PROLOGUF DS 0H
000000	90EC D09C	0000C		9	STM 14,12,12(13)
000004	18CF		0000C	10	USING DEBUT,12
000006	50D0 C078	00078		11	LR 12,15
00000A	41D0 C074	00074		12	ST 13,SAVE+4
				13	LA 13,SAVE
00000E	4150 C000	00000		15	LA INC2,0
000012	4140 0005	00005		16	LA INC1,L'CH1
000016				17	BCL DS 0H
000016	4334 C067	00067		18	IC WORK,CH1-1(IND1)
00001A	4236 C06D	0006D		19	STC WORK,CH2(IND2)
00001E	4155 00C1	00001		20	LA INC2,1(IND2)
000022	4640 C016	00016		21	BCT INC1,BCL

BCTR R,0 تمرين 4.12 -

BALR R,0 تمرين 5.12 -

XC ZONE,ZONE منطقة بطول L تمرين 1.13 -
XR R1,R1 (مرفص)
NI OCTET,X'00' بايت

تمرين 2.13 - لنفترض التعليمة في العنوان INSTR . إذن يوجد كود الطول في INSTR + 1 (تعليمة بنسق SS) .

NI (إعادة تصفير)

OI (حيث XX هو الطول ناقص واحد)

علينا أن نتذكر أنه ، بالنسبة للتعليقات من النوع SS ، الطول المؤول هو الطول ناقص واحد .

XC ZONE1(L),ZONE2 تمرين 3.13 -
XC ZONE2(L),ZONE1
XC ZONE1(L),ZONE2

XR R1,R2
XR R2,R1
XR R1,R2

PACK OCTET,OCTET أو UNPK OCTET,OCTET

تمرين 4.13 - كما في التمرين 6.11 ، نستعمل كون العمليات مع التعليقات MVC ، CLC ... تجري بايت بعد بايت مع انتشار من اليسار إلى اليمين .

```

CLI    ZONE,X'00' (مقارنة أول بايت مع 0)
BNE    NONZERO
CLC    ZONE+1(L'ZONE-1),ZONE
BNE    NONZERO
BE     ZERO

```

التعريف إلى « الفراغ blank » يتم عبر المقارنة مع X'40' .

تمرين 5.13 -

```

XI     ++5,X'F0'
NOP    ETIQ

```

تمرين 6.13 -

```

NOP    ETIQ
XI     --3,X'F0'

```

تمرين 7.13 . INCRE و REF يشكّان مرصفاً مزدوجاً يحتوي الزيادة والمرجع بالنسبة للتعليمات BXLE و NOMBRE هو عنوان العدد. أما PTR فهو مرصيف مصوّب (مؤشّر) .

			NOMBRE					
			+					
			F0	F0	F0	F4	F2	F1
			↑PTR			↑REF		
TEST	LA	PTR,NOMBRE						
	LA	REF,L'NOMBRE-1(PTR)						
	LA	INCRE,1						
	CLI	O(PTR),C'0'						
	BNE	SUITE						
	MVI	O(PTR),C'1'						
SUITE	BXLE	PTR,INCRE,TEST						
						

تمرين 8.13 .

```

INDIC  DC    X'00'
INDEC  EQU   X'80'
INDECR EQU   X'40'
INDWAIT EQU   X'20'
-----
OI     INDIC,INDWAIT
OI     INDIC,INDEC+INDWAIT
NI     INDIC,X'FF'-(INDEC+INDECR)
-----
TM     INDIC,INDWAIT
BO     ALPHA
-----
TM     INDIC,INDEC+INDWAIT
BO     BETA
BM     GAMMA
BZ     DELTA

```

مع هذا الحل فإن التعديل المتعلق بـ INDEC يُترجم بواسطة :

INDLEC EQU X'01'

لا تتأثر أي تعليمة تحديد موضوع أو اختبار . والأمر لا يكون كذلك إن نحن لم نستعمل EQU لتحديد المؤشرات الثنائية ، فحينئذٍ لكان الكود مجمداً بسبب ظهور القيم 'X'80 . . . في قلب التعليمات نفسها . من جهة أخرى فإن هذه التقنية تحوّل التعليمات لأن تصبح موثقة ذاتياً .

تمرين 1.14 -
SLL R,32
SRL R,32 أو

تمرين 2.14 -
SLA R,3 : 2³ يكتب الضرب بـ
SRA R,4 : 16 القسمة على

ترافق القسمة عملية بتر (قطع) . والتمثيل بالمكمل إلى 2 يجعل 15/2 +
تعطي 7 في حين أن 15/2 - تعطي -8 .

تمرين 3.14 -
SLDA R,0
ZERO
R هو مصرف مزدوج

تمرين 4.14 . أثناء عملية إزاحة دائرية إلى اليسار نحاول إعادة إدخال كل بته خارجة في جهة اليمين . العمل يتم على مصرف مزدوج . بعد تصفير مصرف اليسار نجري إزاحة مزدوجة بشكل يسمح بأن نجد من جديد في مصرف اليسار البتات المفقودة في مصرف اليمين . ونتيح لنا تعليمة أو (OR) بإعادة وضعها في مصرف اليمين . هنا نجري إزاحة دائرية من أربعة مواقع على المصرف 7 .

تصفير
SLL 6,32
SLDL 6,4
OR 7,6

تمرين 1.18 -
نستعمل تعليمة TR « بالقلب »

TR	CLE,ARTICLE	
---	----	
ARTICLE	DC	CL10 'ABCDEFGH1J' (فقرة)
CLE	DS	OCL5 (مفتاح)
	DC	HL1'5,6,7,1,2'

تمرين 2.18 -

نستعمل التعليمة TR

TR CHAINE(8),TABLE

DC C'0123456789ABCDEF'

(جدول)

DC 2F

(سلسلة)

ملحقات

جدول تكويد السمات
جدول أبجدي للتعليقات
أوامر المؤول
مميزات الثوابت
كود حرفي (تذكيري) موسّع

جدول تكميد السمات

بطاقة مفتوحة	سمة مطبوعة	تذكيري حرفي	سادس عشري	عشري	بطاقة مفتوحة	سمة مطبوعة	تذكيري حرفي	سادس عشري	عشري
لا تقييد	بياض	STH	40	64	12-0-9-8-1			00	0
12-0-9-1		LA	41	65	12-9-1			01	1
12-0-9-2		STC	42	66	12-9-2			02	2
12-0-9-3		IC	43	67	12-9-3			03	3
12-0-9-4		EX	44	68	12-9-4		SPM	04	4
12-0-9-5		BAL	45	69	12-9-5		BALR	05	5
12-0-9-6		BCT	46	70	12-9-6		BCTR	06	6
12-0-9-7		BC	47	71	12-9-7		BCR	07	7
12-0-9-8		LH	48	72	12-9-8		SSK	08	8
12-8-1		CH	49	73	12-9-8-1		ISK	09	9
12-8-2	(نقطة)	AH	4A	74	12-9-8-2		SVC	0A	10
12-8-3		SH	4B	75	12-9-8-3			0B	11
12-8-4		MH	4C	76	12-9-8-4			0C	12
12-8-5	(+)		4D	77	12-9-8-5		MVCL	0D	13
12-8-6		CVD	4E	78	12-9-8-6			0E	14
12-8-7	&	CVB	4F	79	12-9-8-7		CLCL	0F	15
12		ST	50	80	12-11-9-8-1		LPR	10	16
12-11-9-1			51	81	11-9-1		LNR	11	17
12-11-9-2			52	82	11-9-2		LTR	12	18
12-11-9-3			53	83	11-9-3		LCR	13	19
12-11-9-4		N	54	84	11-9-4		NR	14	20
12-11-9-5		CL	55	85	11-9-5		CLR	15	21
12-11-9-6		O	56	86	11-9-6		OR	16	22
12-11-9-7		X	57	87	11-9-7		XR	17	23
12-11-9-8		L	58	88	11-9-8		LR	18	24
11-8-1		C	59	89	11-9-8-1		CR	19	25
11-8-2		A	5A	90	11-9-8-2		AR	1A	26
11-8-3	s	S	5B	91	11-9-8-3		SR	1B	27
11-8-4	*	M	5C	92	11-9-8-4		MR	1C	28
11-8-5)	D	5D	93	11-9-8-5		DR	1D	29
11-8-6		AL	5E	94	11-9-8-6		ALR	1E	30
11-8-7	-	SL	5F	95	11-9-8-7		SLR	1F	31
11		STD	60	96	11-0-9-8-1		LPDR	20	32
0-1	/		61	97	0-9-1		LNDR	21	33
11-0-9-2			62	98	0-9-2		LTDR	22	34
11-0-9-3			63	99	0-9-3		LCDR	23	35
11-0-9-4			64	100	0-9-4		HDR	24	36
11-0-9-5			65	101	0-9-5		LRDR	25	37
11-0-9-6			66	102	0-9-6		MXR	26	38
11-0-9-7		MXD	67	103	0-9-7		MXDR	27	39
11-0-9-8		LD	68	104	0-9-8		LDR	28	40
0-8-1		CD	69	105	0-9-8-1		CDR	29	41
12-11		AD	6A	106	0-9-8-2		ADR	2A	42
0-8-3		SD	6B	107	0-9-8-3		SDR	2B	43
0-8-4	%	MD	6C	108	0-9-8-4		MDR	2C	44
0-8-5	-	DD	6D	109	0-9-8-5		DDR	2D	45
0-8-6	>	AW	6E	110	0-9-8-6		AWR	2E	46
0-8-7	?	SW	6F	111	0-9-8-7		SWR	2F	47
12-11-0		STE	70	112	12-11-0-9-8-1		LPER	30	48
12-11-0-9-1			71	113	9-1		LNER	31	49
12-11-0-9-2			72	114	9-2		LTER	32	50
12-11-0-9-3			73	115	9-3		LCER	33	51
12-11-0-9-4			74	116	9-4		HER	34	52
12-11-0-9-5			75	117	9-5		LRER	35	53
12-11-0-9-6			76	118	9-6		AXR	36	54
12-11-0-9-7			77	119	9-7		SKR	37	55
12-11-0-9-8		LE	78	120	9-8		LER	38	56
8-1		CE	79	121	9-8-1		CER	39	57
8-2		AE	7A	122	9-8-2		AER	3A	58
8-3	*	SE	7B	123	9-8-3		SER	3B	59
8-4		ME	7C	124	9-8-4		MER	3C	60
8-5		DE	7D	125	9-8-5		DER	3D	61
8-6		AU	7E	126	9-8-6		AUR	3E	62
8-7	"	SU	7F	127	9-8-7		SUR	3F	63

جدول تكويد السيات

بطاقة مشفرة	رمز مطبوعة	حرفي تذكيري	سادس عشري	عشري	بطاقة مشفرة	رمز مطبوعة	حرفي تذكيري	سادس عشري	عشري
128	80	SSM		12-0-8-1	192	C0			12-0
129	81			12-0-1	193	C1		A	12-1
130	82	LPSW		12-0-2	194	C2		B	12-2
131	83			12-0-3	195	C3		C	12-3
132	84	WRD		12-0-4	196	C4		D	12-4
133	85	RDD		12-0-5	197	C5		E	12-5
134	86	BXH		12-0-6	198	C6		F	12-6
135	87	BKLE		12-0-7	199	C7		G	12-7
136	88	SRL		12-0-8	200	C8		H	12-8
137	89	SLL		12-0-9	201	C9		I	12-9
138	8A	SRA		12-0-8-2	202	CA			12-0-9-8-2
139	8B	SLA		12-0-8-3	203	CB			12-0-9-8-3
140	8C	SRDL		12-0-8-4	204	CC			12-0-9-8-4
141	8D	SLDL		12-0-8-5	205	CD			12-0-9-8-5
142	8E	SRDA		12-0-8-6	206	CE			12-0-9-8-6
143	8F	SLDA		12-0-8-7	207	CF			12-0-9-8-7
144	90	STM		12-11-8-1	208	D0			11-0
145	91	TM		12-11-1	209	D1	MVN	J	11-1
146	92	MVI		12-11-2	210	D2	MVC	K	11-2
147	93	TS		12-11-3	211	D3	MVZ	L	11-3
148	94	NI		12-11-4	212	D4	NC	M	11-4
149	95	CLI		12-11-5	213	D5	CLC	N	11-5
150	96	OI		12-11-6	214	D6	OC	O	11-6
151	97	XI		12-11-7	215	D7	XC	P	11-7
152	98	LM		12-11-8	216	D8		Q	11-8
153	99			12-11-9	217	D9		R	11-9
154	9A			12-11-8-2	218	DA			12-11-9-8-2
155	9B			12-11-8-3	219	DB			12-11-9-8-3
156	9C	SIO		12-11-8-4	220	DC	TR		12-11-9-8-4
157	9D	TIO		12-11-8-5	221	DD	TRT		12-11-9-8-5
158	9E	HIO		12-11-8-6	222	DE	ED		12-11-9-8-6
159	9F	TCH		12-11-8-7	223	DF	EDMK		12-11-9-8-7
160	A0			11-0-8-1	224	E0			0-8-2
161	A1			11-0-1	225	E1			11-0-9-1
162	A2			11-0-2	226	E2		S	0-2
163	A3			11-0-3	227	E3		T	0-3
164	A4			11-0-4	228	E4		U	0-4
165	A5			11-0-5	229	E5		V	0-5
166	A6			11-0-6	230	E6		W	0-6
167	A7			11-0-7	231	E7		X	0-7
168	A8			11-0-8	232	E8		Y	0-8
169	A9			11-0-9	233	E9		Z	0-9
170	AA			11-0-8-2	234	EA			11-0-9-8-2
171	AB			11-0-8-3	235	EB			11-0-9-8-3
172	AC	STNSM		11-0-8-4	236	EC			11-0-9-8-4
173	AD	STOSM		11-0-8-5	237	ED			11-0-9-8-5
174	AE	SIGF		11-0-8-6	238	EE			11-0-9-8-6
175	AF	MC		11-0-8-7	239	EF			11-0-9-8-7
176	B0			12-11-0-8-1	240	F0	SRP	0	0
177	B1	LRA		12-11-0-1	241	F1	MVO	1	1
178	B2			12-11-0-2	242	F2	PACK	2	2
179	B3			12-11-0-3	243	F3	UNPK	3	3
180	B4			12-11-0-4	244	F4		4	4
181	B5			12-11-0-5	245	F5		5	5
182	B6	STCTL		12-11-0-6	246	F6		6	6
183	B7	LCIL		12-11-0-7	247	F7		7	7
184	B8			12-11-0-8	248	F8	ZAP	8	8
185	B9			12-11-0-9	249	F9	CP	9	9
186	BA	CS		12-11-0-8-2	250	FA	AP		12-11-0-9-8-2
187	BB	CDS		12-11-0-8-3	251	FB	SP		12-11-0-9-8-3
188	BC			12-11-0-8-4	252	FC	MP		12-11-0-9-8-4
189	BD	CLM		12-11-0-8-5	253	FD	DP		12-11-0-9-8-5
190	BE	STCM		12-11-0-8-6	254	FE			12-11-0-9-8-6
191	BF	ICM		12-11-0-8-7	255	FF			12-11-0-9-8-7

جدول أبجدي للتعليمات

النسق	منطقة العوامل	Format	منطقة العوامل
RR	R_1, R_2	SI	$D_1(B_1), I_2$
RR-M	M_1, R_2		
RR-1	R_1	S	$D_2(B_2)$
RR-I	I_1		
RX	$R_1, D_2(X_2, B_2)$	SS-1	$D_1(L, B_1), D_2(B_2)$
RX-M	$M_1, D_2(X_2, B_2)$	SS-2	$D_1(L_1, B_1), D_2(L_2, B_2)$
		SS-3	$D_1(L_1, B_1), D_2(B_2), I_3$
RS	$R_1, R_3, D_2(B_2)$	R, X مرادف D إزاحة M قطاع بأربعة بتات I قيمة فورية L طول	
RS-M	$R_1, M_3, D_2(B_2)$		

الذالة (الوظيفة)	حرفي تذكيري	COP سادس عشري	النسق	محدد موضع CC
Add	AR	1A	RR	*
Add	A	5A	RX	*
Add Decimal	AP	FA	SS-2	*
Add Halfword	AH	4A	RX	*
Add Logical	ALR	1E	RR	*
Add Logical	AL	5E	RX	*
AND	NR	14	RR	*
AND	N	54	RX	*
AND	NI	94	SI	*
AND	NC	D4	SS-1	*
Branch and Link	BALR	05	RR	
Branch and Link	BAL	45	RX	
Branch on Condition	BCR	07	RR-M	
Branch on Condition	BC	47	RX-M	
Branch on Count	BCTR	06	RR	
Branch on Count	BCT	46	RX	
Branch on Index High	BXH	86	RS	
Branch on Index Low or Equal	BXLE	87	RS	
Compare	CR	19	RR	*
Compare	C	59	RX	*
Compare and Swap	CS	BA	RS	*
Compare Decimal	CP	F9	SS-2	*
Compare Double and Swap	CDS	BB	RS	*
Compare Halfword	CH	49	RX	*
Compare Logical	CLR	15	RR	*
Compare Logical	CL	55	RX	*
Compare Logical	CLC	D5	SS-1	*
Compare Logical	CLI	95	SI	*
Compare Logical Characters under Mask	CLM	BD	RS-M	*
Compare Logical Long	GLCL	0F	RR	*
Convert to Binary	CVB	4F	RX	
Convert to Decimal	CVD	4E	RX	

الدالة (الوظيفة) Fonction	حرفي تذكيري Mnemo- nique	سادس عشري hexa- derimal	النسق Format	محدد موضع CC
Divide	DR	1C	RR	
Divide	D	5D	RX	
Divide Decimal	DP	FD	SS-2	
Edit	ED	DE	SS-1	*
Edit and Mark	EDMK	DF	SS-1	*
Exclusive OR	XR	17	RR	*
Exclusive OR	X	57	RX	*
Exclusive OR	XI	97	SI	*
Exclusive OR	XC	D7	SS-1	*
Execute	EX	44	RX	
Insert Character	IC	43	RX	
Insert Characters under Mask	ICM	BF	RS-M	*
Load	LR	18	RR	
Load	L	58	RX	
Load Address	LA	41	RX	
Load and Test	LTR	12	RR	*
Load Complement	LCR	13	RR	*
Load Halfword	LH	48	RX	
Load Multiple	LM	98	RS	
Load Negative	LNP	11	RR	*
Load Positive	LPR	10	RR	*
Monitor Call	MC	AF	SI	
Move	MVI	92	SI	
Move	MVC	D2	SS-1	
Move Long	MVCL	0E	RR	*
Move Numerics	MVN	D1	SS-1	
Move with Offset	MVO	F1	SS-2	
Move Zones	MVZ	D3	SS-1	
Multiply	MR	1C	RR	
Multiply	M	5C	RX	
Multiply Decimal	MP	FC	SS-2	
Multiply Halfword	MH	4C	RX	
OR	GR	16	RR	*
OR	O	56	RX	*
OR	OI	96	SI	*
OR	OC	D6	SS-1	*
Pack	PACK	F2	SS-2	
Set Program Mask	SPM	04	RR-1	
Shift and Round Decimal	SRP	F0	SS-3	*
Shift Left Double	SLDA	8F	RS	*
Shift Left Double Logical	SLDL	8D	RS	
Shift Left Single	SLA	8B	RS	*
Shift Left Single Logical	SLL	89	RS	
Shift Right Double	SRDA	8E	RS	*
Shift Right Double Logical	SRDL	8C	RS	
Shift Right Single	SRA	8A	RS	*
Shift Right Single Logical	SRL	88	RS	
Store	ST	50	RX	
Store Character	STC	42	RX	
Store Characters under Mask	STCM	BE	RS-M	
Store Clock	STCK	B205	S	*
Store Halfword	STH	40	RX	
Store Multiple	STM	90	RS	

Subtract	SR	1B	RR	*
Subtract	S	5B	RX	*
Subtract Decimal	SP	FB	SS-2	*
Subtract Halfword	SH	4B	RX	*
Subtract Logical	SLR	1F	RR	*
Subtract Logical	SL	5F	RX	*
Supervisor Call	SVC	0A	RR-1	
Test and Set	TS	93	S	*
Test under Mask	TM	91	SI	*
Translate	TR	DC	SS-1	
Translate and Test	TRT	DD	SS-1	*
Unpack	UNPK	F3	SS-2	
Zero and Add Decimal	ZAP	F8	SS-2	*

تعليمات حسابية بالفاصلة المتحركة

Add Normalized, Extended	AXR	36	RR	*
Add Normalized, Long	ADR	2A	RR	*
Add Normalized, Long	AD	6A	RX	*
Add Normalized, Short	AER	3A	RR	*
Add Normalized, Short	AE	7A	RX	*
Add Unnormalized, Long	AWR	2E	RR	*
Add Unnormalized, Long	AW	6E	RX	*
Add Unnormalized, Short	AUR	3E	RR	*
Add Unnormalized, Short	AU	7E	RX	*
Compare, Long	CDR	29	RR	*
Compare, Long	CD	69	RX	*
Compare, Short	CER	39	RR	*
Compare, Short	CE	79	RX	*
Divide, Long	DDR	2D	RR	
Divide, Long	DD	6D	RX	
Divide, Short	DER	3D	RR	
Divide, Short	DE	7D	RX	
Halve, Long	HDR	24	RR	
Halve, Short	HER	34	RR	
Load and Test, Long	LTDR	22	RR	*
Load and Test, Short	LTER	32	RR	*
Load Complement, Long	LCDR	23	RR	*
Load Complement, Short	LCER	33	RR	*
Load, Long	LDR	28	RR	
Load, Long	LD	68	RX	
Load Negative, Long	LNDR	21	RR	*
Load Negative, Short	LNER	31	RR	*
Load Positive, Long	LPDR	20	RR	*
Load Positive, Short	LPER	30	RR	*
Load Rounded, Extended Long	LRDR	25	RR	
Load Rounded, Long to Short	LRER	35	RR	
Load, Short	LER	38	RR	
Load, Short	LE	78	RX	
Multiply, Extended	MXR	26	RR	
Multiply, Long	MDR	2C	RR	
Multiply, Long	MD	6C	RX	
Multiply, Long/Extended	MXDR	27	RR	
Multiply, Long/Extended	MXD	67	RX	
Multiply, Short	MER	3C	RR	
Multiply, Short	ME	7C	RX	
Store, Long	STD	60	RX	

Store, Short	STE	70	RX	
Subtract Normalized, Extended	SXR	37	RR	*
Subtract Normalized, Long	SDR	2B	RR	*
Subtract Normalized, Long	SD	6B	RX	*
Subtract Normalized, Short	SER	3B	RR	*
Subtract Normalized, Short	SE	7B	RX	*
Subtract Unnormalized, Long	SWR	2F	RR	*
Subtract Unnormalized, Long	SW	6F	RX	*
Subtract Unnormalized, Short	SUR	3F	RR	*
Subtract Unnormalized, Short	SU	7F	RX	*

أوامر المؤول

تعريف المعطيات	DC DS CCW	Assemblage conditionnel	تأويل مشروط	MACRO MNOTE MEXIT MEND
تقطيع	START CSECT DSECT COM ENTRY EXTRN			ACTR AGO AIF ANOP GBLA GBLB GBLC LCLA LCLB LCLC SETA SETB SETC
تعريف المراصف القاعدية	USING DROP			
مراقبة اللائحة	TITLE EJECT SPACE PRINT			
مراقبة البرنامج	EOU ORG LTORG CNOP END COPY PUNCH REPRO ISEQ ICTL PUSH POP OPSYN			

مميزات الثوابت

النوع	الطول الضمي	حد الإصطفاف	يتميز بـ	يتر أو ملء إلى
C	-	بايتة	سبات	اليمن
X	-	بايتة	أرقام سادس عشرية	اليسار
B	-	بايتة	أرقام ثنائية	اليسار
F	4	كلمة	أرقام عشرية	اليسار
H	2	نصف كلمة	أرقام عشرية	اليسار
E	4	كلمة	أرقام عشرية	اليمن
D	8	كلمة مزدوجة	أرقام عشرية	اليمن
L	16	كلمة مزدوجة	أرقام عشرية	اليمن
P	-	بايتة	أرقام عشرية	اليسار
Z	-	بايتة	أرقام عشرية	اليسار
A	4	كلمة	تعبير	اليسار
Y	2	نصف كلمة	تعبير	اليسار
S	2	نصف كلمة	تعبير	-
V	4	كلمة	مز قابل للنقل	اليسار

الكود الحرفي موسع

القناع	التعليمة المولدة	المعنى ...	كود العملية الحرفي
1111	BC 15,... BCR 15,...	تفريع غير مشروط	B ... BR ...
0000	BC 0,... BCR 0,...	لا عملية	NOP ... NOPR ...
... بعد تعليقات المقارنة			
تفريع إذا كان :			
0010	BC 2,... BCR 2,...	(*) المتأثر 2 > المتأثر 1 (*) Op	BH ... BHR ...
0100	BC 4,... BCR 4,...	" < "	BL ... BLR ...
1000	BC 8,... BCR 8,...	" = "	BE ... BER ...
1101	BC 13,... BCR 13,...	" ≤ "	BNH ... BNHR ...
1011	BC 11,... BCR 11,...	" ≥ "	BNL ... BNLR ...
0111	BC 7,... BCR 7,...	" ≠ "	BNE ... BNER ...
... بعد التعليقات الحسابية			
تفريع إذا كانت النتيجة			
0001	BC 1,... BCR 1,...	فيض عن السعة ...	BO ... BOR ...
0010	BC 2,... BCR 2,...	... > 0	BP ... BPR ...
0100	BC 4,... BCR 4,...	... < 0	BM ... BMR ...
1101	BC 13,... BCR 13,...	... ≤ 0	BNP ... BNPR ...
1011	BC 11,... BCR 11,...	... ≥ 0	BNM ... BNMR ...
0111	BC 7,... BCR 7,...	... ≠ 0	BNZ ... BNZR ...
1000	BC 8,... BCR 8,...	... = 0	BZ ... BZR ...

(*) المقصود هما المتأثران 1 و 2 في تعليمة المقارنة .

ملاحظة : الكود الحرفي التذكيري المنتهي بحرف "2" بولّد تعليمات من النسق RR . المرصف المذكور يحتوي على عنوان التفريع .

مثلاً : BR 3 تفريع غير مشروط إلى العنوان الواقع في المرصف 3 .

B ALPHA تفريع غير مشروط إلى العنوان ALPHA .

ترجمة الملاحظات الواردة في بعض البرامج الموجودة في الكتاب

الصفحة	الملاحظة	السطر
69	5 ثوابت سهات . لا يوجد اصطفااف خاص . الطول 256	
	6 تأطير إلى اليسار . بقدر إلى اليمين	
	8 بتر إلى اليمين .	
	9 تأطير إلى اليسار تكمله فراغات .	
	10 توليد فاصلة عليا واحدة .	
	11 نفس الملاحظة	
	12 تكرار وبت	
	15 ثوابت سادس عشرية . تأطير إلى اليمين . بتر إلى اليسار .	
	16 طول ضمني	
	17 طول ظاهر .	
	18 بتر .	
	21 ثوابت ثنائية . الطول الأقصى 256 بايت تأطير إلى اليمين .	
	22 تكمله أصفار إلى اليسار . اصطفااف على البايته .	
	23 ثنائي	
	24 بتر إلى اليسار .	
	25 نر .	
	26 تكرار .	
	29 ثوابت بالفاصلة الثابتة على كلمة (F) أو نصف كلمة (H) .	
	30 اصطفااف على الكلمة أو نصف الكلمة . عندما يكون الطول	
	31 محذرا لا يعود هناك اصطفااف . الثابتة هي بالنظام العشري	
	34 إزاحة 3 بتات إلى اليسار (8 *)	
	36 إزاحة 3 بتات إلى اليمين (/8)	
	39 مدور أعلى	
	40 مدور أصغر .	
	42 تعديل الطول LONG والاصطفااف ALIGN .	
	43 إزاحة بتتين إلى اليسار .	
70	49 ثوابت بالفاصلة المتحركة والدقة البسيطة . اصطفااف على الكلمة	
	50 تأطير إلى اليمين . لا بتر . القيمة مدورة .	
	51 الطول الضمني 4 بايتات .	

- 52 بفاصلة متحركة
57 ثوابت بالفاصلة المتحركة وبالدقة المزدوجة
58 اصطفااف على الكلمة المزدوجة . تأطير إلى اليمين . لا بتر
59 القيمة مدوّرة . الطول الضمني 8 بايتات
66 ثوابت بالفاصلة المتحركة وبالدقة الرباعية
67 اصطفااف على الكلمة المزدوجة . الطول الضمني 16 بايتة .
68 لا بتر . القيمة «تعدّورة» . أس من -85 إلى +75 .
73 ثوابت عشرية . الطول الأقصى يبلغ
74 16 بايتة . الإشارة تقع في الربع الأيسر
75 من البايته اليمنى الأخيرة . تأطير إلى اليمين . بتر إلى اليسار .
76 $X'F$ أو $X'C$ في موقع الإشارة يُعتبران مثل +
77 $X'D$ أو $X'E$ في موقع الإشارة يُعتبران مثل -
78 لا يتم ترجمة الفاصلة العشرية أبداً إلى الثنائي .
79 تأطير إلى اليمين . بتر إلى اليسار .
80 الطول الضمني .
82 بتر إلى اليسار
86 الثوابت العشرية المكثفة (Packed)
87 نفس قواعد الثوابت السابقة .
88 تقع الإشارة في الربع الأيمن الأخير .

- 2 رمز خارجي
6 رمز قابل للنقل
9 ثوابت عنوان من النوع A
10 تُكتب DC A (تعبير مطلق أو قابل للنقل)
11 اصطفااف على الكلمة . الطول الضمني 4 بايتات .
12 الأطوال الظاهرة الممكنة هي من 1 إلى 4 بايتات .
13 بتر إلى اليسار . ممكن التحديد في كود حرفي .
18 طول ظاهر
20 رمز خارجي
23 ثوابت عنوان من النوع Y
24 تُكتب DC Y (تعبير مطلق أو قابل للنقل)
25 اصطفااف على نصف الكلمة . الطول الضمني نصف كلمة .
26 الأطوال الظاهرة الممكنة هي من 1 أو 2 بايتة .
27 بتر إلى اليسار . ممكن التحديد في كود حرفي .
29 لاحظوا أنّ النجمتين
30 تساويان B و B + 2
31 الطول الظاهر
32 بتر إلى اليسار
35 ثوابت عنوان من النوع S
36 تُكتب DC S (تعبير مطلق) .
37 DC S (تعبير قابل للنقل) .
38 DC S (تعبير مطلق) .

73

- 39 مؤولة في نصف كلمة . مصطفة عل . نصف الكلمة .
 40 لا يمكن تحديدها في كود حربي .
 42 القاعدة (Base) 0 ، الإزاحة (Déplacement) = 1024
 43 قاعدة وإزاحة RELOC
 49 ثوابت عنوان من النوع V
 50 تستعمل فقط للعناوين الخارجية من النوع اسم البرنامج NOU-DE-PROG .
 51 تكتب V DC (رمز خارجي قابل للنقل)
 52 لا يرد الرمز القابل للنقل في أمر خارجي .
 53 الطول الضمني 4 بايتات . معدل الطول = 3 أو 4 .
 54 اصطفااف عل حدّ كلمة ، بإمكانه أن يظهر في كود حربي .
 55 يولّد المؤؤل كلمة صفر .

79

- 3 متالية الدخول
 4 و5 حفظ المرافف من شحن مصرف القاعدة
 6 R12 = مصرف القاعدة
 7 البرنامج المنادي
 14 اصطفااف كلمة
 18 (1) القاعدة 12 ظاهرة
 19 (2) القاعدة 12 ظاهرة
 21 و22 و23 كلّ التعليقات من (3) حتّى (7) تشحن X'89ABCDEF في المصرف 3 . الكتابة (3) هي الرجيلة
 المستقلة عن مكان ALPHA بالنسبة إلى عنوان القاعدة .
 24 و25 (3) استعمال تعبير قابل للنقل . قاعدة ضمنية .
 26 (4) تعليمة تماثل رمزا مطلقا .
 30 (7) استعمال كود حربي
 32 (8) "8" هي عبارة عن إزاحة
 24 (9) خطأ اصطفااف
 36 (10) "12" هي عبارة عن إزاحة
 37 (11) "12" هي عبارة عن مصرف قاعدي
 38 (12) خطأ في النحو
 39 (13) خطأ في النحو
 40 (14) خطأ في النحو
 41 (15) 12 هي عبارة عن مؤشّر

111

- 3 مؤشّر (مصوب) إلى عنصر من TAB
 4 مصرف إضافة لـ BXLE
 5 مصرف مرجع لـ BXLE
 6 مصرف عمل
 11 القاعدة = المصرف 12
 16 تصفير (إعداد)
 23 طول الكلمة

111

- 47 حلقة مسح الجدول
- 48 و49 في حال عدم التبديل يتم فرز (ترتيب) TAB
- 52 تصنيف المؤشر
- 53 تصنيف مرجع BXLE
- 56 عنصر ايسر في مصرف العمل
- 57 مقارنة
- 60 تبديل
- 62 تحديد موقع INDIC
- 101 منطقة المعطيات
- 102 عدد عناصر TAB
- 105 إعداد INDIC

115

- 39 مؤشر بداية الجدول الثانوي
- 40 مؤشر نهاية الجدول الثانوي
- 41 مؤشر المتصف والرتبة
- 42 مصرف العمل
- 43 طول العنصر
- 62 عدد عمليات التكرار في البرنامج
- 76 إعداد
- 80 حساب عنوان العنصر الوسط (المتصف)
- 84 قسمة على $L * 2$
- 85 (PTRELEM) = عدد العناصر في الجدول الثانوي
- 87 إذا 0 فرغم حتى 1
- 89 ضرب $>$
- 91 مقارنة
- 92 تفرع إذا كان $(MOT) < ELEM$
- 93 تفرع إذا كان $(MOT) > ELEM$
- 95 وجدنا العنصر حساب رتبة العنصر $(MOT) =$
- 98 قسمة على الطول
- 100 طباعة الرتبة والقيمة

116

- 129 لم نجده
- 153 منطقة المعطيات
- 154 عدد كلمات الجدول
- 155 طول العنصر

160

- 3 حفظ مراصف المتادي
- 5 تعريف وشحن مصرف القاعدة
- 6 تأخذ المرصف 12 كقاعدة
- 7 عنوان PROG في 12
- 8 حفظ R13 في المنطقة SAVE AREA من البرنامج .

الصفحة

- 10 و 11 حفظ عنوان المنطقة SAVE AREA من هذا البرنامج في المنطقة AREA من الناحي 160
- 16 تعريف المنطقة SAVE AREA
- 22 متتالية نداء PROGK
- 29 متتالية العودة إلى PROGI

فهرست

الموضوع	الصفحة
تقديم	5
تمهيد	7

القسم الأول : عموميات

1 - الآلة البسيطة	9
2 - تكويد المعلومات	20
تمارين	35
3 - العنوان المطلقة ، العنوان النسبية	36
4 - هيكلية الحاسبات IBM 360/370	41
5 - لغة الآلة	45
6 - لغة المؤول	51

القسم الثاني 360 / 370

7 - العناصر الأساسية	59
8 - توجيهات تعريف الرموز	67
تمارين	75
9 - كتابة العناوين بلغة المؤول	76
10 - التعليمات بلغة المؤول ، عموميات	81
11 - الحساب بفاصلة ثابتة والحركات	84
تمارين	92
12 - التفريعات	93
تمارين	98

99	13 - العمليات المنطقية
104	تمارين
106	14 - عمليات الإزاحة
109	تمارين
110	15 - مسائل
117	16 - الحساب العشري
120	17 - الحساب بقاصلة متحركة
123	18 - تعليمات التحويل والتمثيل
127	تمارين
129	19 - الانقطاع والادخال والاخراج
138	20 - الأوامر المتعلقة بالعنونة وتركيبية المرنجج
152	21 - البرامج الثانوية
162	22 - التأويل المشروط وماكرو التعليمات
177	23 - نصائح في البرمجة
181	حلول التمارين
189	ملحقات
190	جدول تكويد السمات
192	جدول أبجدي للتعليمات
195	أوامر المؤول
196	مميزات الثوابت
197	الكود الحرفي موسع

هذا الكتاب

تعتبر لغة المؤول (الأسمبلر) من العناصر الأساسية في التفكير حول طريقة البرمجة بإحدى اللغات المتطورة فهي تتيح لنا فهماً مفصلاً لأليات الحاسب وليس بالإمكان الاستغناء عنها في إعداد المعلومات .

وتتجلى ضرورة استعمال لغة المؤول ، بالرغم من قوة اللغات المتطورة ، عندما يوجد إلزات بالنسبة لفترات الإجابة (بعض البرامج الكبيرة ، أنظمة التشغيل ، المصروفات ، الوقت الحقيقي ، ...) أو بالنسبة لحجم الذاكرة (الحاسبات الصغيرة والمتوسطة) ، أو أيضاً إلزات تعود إلى عدم كفاية إمكانات البرامج (فورتران ، باسيك) .

من جهة أخرى ، سوف يجد مستعملو الميكرومعلوماتية في تطبيق لغة المؤول حلاً ممتازاً لما يعترضهم من مشاكل .

يتوجه هذا الكتاب إلى الطلاب والممارسين الذين يرغبون بتعميق معرفتهم في مجال المعلوماتية . وهو يتكون من فصول قصيرة وبتدئ انطلاقة من ملاحظات بسيطة جداً على حاسبة الجيب ، بشكل يقود معه القارئ شيئاً فشيئاً ، لا سيما بفضل التمارين المحلولة والمفاهيم الأساسية في بنية الآلة ، إلى دراسة المؤول والماكرو - لغة . ولا شك أنه بالإمكان استعماله كمرجع ولتدريس متعلق بسلسلة الآلات المعتمدة كأمثلة (سلاسل 3000 ، 4000 ، IBM370) ولكنه وضع كي يكون دليلاً عاماً يوجه بطريقة سليمة أي برمجة بلغة المؤول .